THREE FULL PRODUCTS ON YOUR COVER CD • PLUS! WIN £1500-WORTH OF SOFTWARE

COMPUTER arts projects

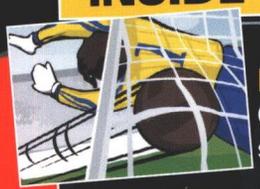
THE IN-DEPTH GUIDE FOR [illegible] CREATIVES

FLASH AND DIRECTOR

Your guide to creating cutting-edge multimedia

120 ACTIONSCRIPT AND LINGO TIPS!

INSIDE THIS ISSUE



FLASH GAMES
Create an exciting penalty shootout game in Flash!

DIGITAL PORTFOLIO
Show off your work in style and get that dream job

SHOCKWAVE 3D
Revealed! The secrets of creating interactive 3D characters

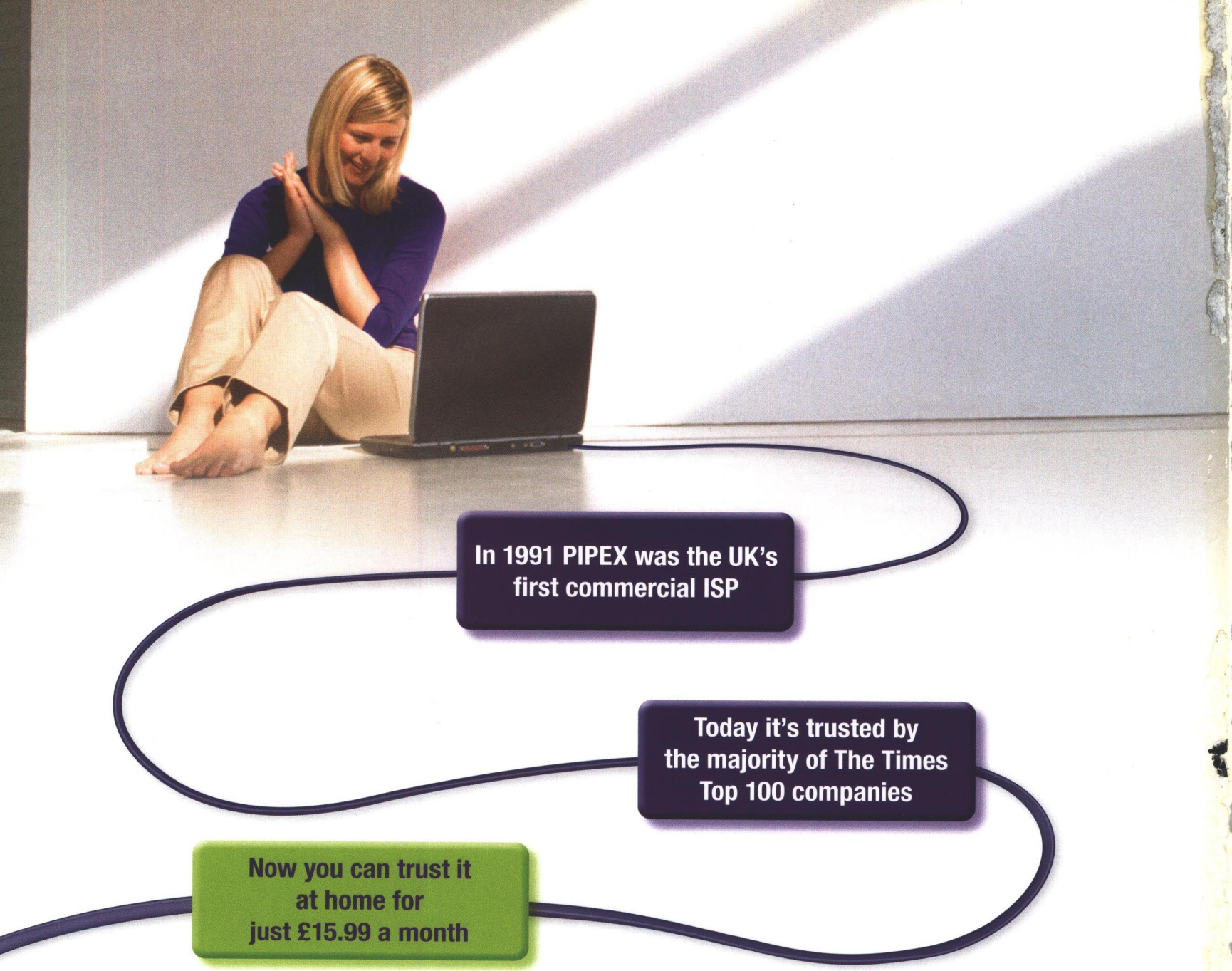
MP3 PLAYER
Use Director to create an MP3 jukebox complete with play-lists

UK edition • Printed in the UK • Outside the UK and Eire £6

ISSUE 61 AUGUST 2004 £6

future publishing
Media with passion

9 771740 900004 08


In 1991 PIPEX was the UK's first commercial ISP
Today it's trusted by the majority of The Times Top 100 companies
Now you can trust it at home for just £15.99 a month

PIPEX professional broadband now available at home

PIPEX has been the leading choice for business for over a decade. Now you can choose it at home too. Industry leading, multi-award winning professional broadband – is it any wonder over 50% of our new customers join PIPEX through word of mouth. So if you want fast, reliable broadband that's up to 20x faster than standard dial-up at home, connect to PIPEX and get the best in broadband.

FROM JUST

£15.99

A MONTH INCLUDING FREE MODEM & FREE SET UP

PIPEX®

Visit www.solo.pipex.net

COMPUTER arts projects

Welcome

Flash and *Director* have become the de facto standard tools for creating cutting-edge multimedia content, whether you're designing for the Web, CD-ROM or DVD. This issue, we've responded to your pleas to bring you some fantastic tutorials to get the most out of these two applications.

Over the course of this issue, you'll find out how to create an eye-catching portfolio in *Flash* (page 14), make yourself an awesome *Flash* penalty shootout game (page 28), build an MP3 player in *Director* (page 84) and learn all the secrets of *Flash* components and Shockwave 3D.

We also bring you 120 ActionScript and Lingo tips. As an added bonus, all the code used in the tips is on your cover CD – so you can copy and paste it into your software and see how it works for yourself.

As if that wasn't enough, there's a fantastic tutorial from our friends at Float Design explaining exactly how to hook up *Flash* and PHP to create an online community. See page 36 for more.

On your cover CD, you get three full products, plus over $1300-worth of *Flash* stock movies. There's also a host of demo software to enhance your multimedia skills.

Simply put, if you want to create cutting-edge multimedia content, then this issue's for you! Let me know what you think, and if you have any ideas for future issues...

Rob Carney, Associate Editor
[e] rob.carney@futurenet.co.uk

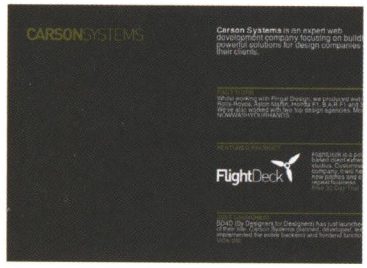
Tell us what topics you'd like to see covered in-depth in *Computer Arts Projects*. Contact us on: **rob.carney@futurenet.co.uk**

Contributing creatives

COVER ILLUSTRATION
Twelve:Ten

Nottingham-based design house Twelve:Ten brought a retro feel to this month's cover illustration, playing on a Flash Gordon theme. The result is a striking illustration that captures the dynamism of creating fantastic multimedia content.

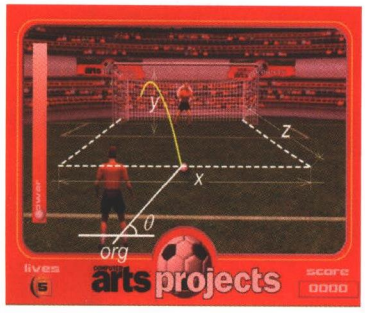
[w] www.twelveten.com



Ryan Carson, Carson Systems

Ryan Carson is a *Flash, Director* and Web applications specialist. As well as writing our *Flash* portfolio and *Director* DVD tutorials, Ryan keeps himself busy developing FlightDeck – a powerful Web-based client extranet.

[w] www.carsonsystems.com



Nik Lever, Catalyst Pictures

Nik Lever is a renowned ActionScript and Lingo expert and has written many books on the subject, including the forthcoming *Flash MX 2004 Games* published by Focal Press. Nik created our penalty shootout game (page 28) as well as contributing our ActionScript and Lingo tips.

[w] www.catalystpics.co.uk

Float Design

We asked Float Design, aka Tim Ketterer and Stephen Lockhart, to design us an online community using *Flash* and PHP. The result is TRUMP TOWN – see it at www.floatdesign.net/trumptown and read more about this up-and-coming company on page 34.

[w] www.floatdesign.net

We're always on the lookout for new artists as well as established creatives – so if you're interested in contributing to the magazine, please send some examples of your work to **[e]richard.llewellyn@futurenet.co.uk**

COMPUTER arts projects

Subscriptions – page 52
Future Publishing Ltd, FREEPOST, BS4900, Somerton, Somerset, TA11 7B
[t] Customer hotline: 0870 444 8455
[f] 01458 271 146
[e] ca.subs@futurenet.co.uk

Editorial [t] 01225 442 244
Rob Carney associate editor
[e] robert.carney@futurenet.co.uk
Ashley Young production editor
[e] casproduction@futurenet.co.uk
Roddy Llewellyn consultant art editor
[e] richard.llewellyn@futurenet.co.uk
Matt Gallimore senior CD editor
[e] matt.gallimore@futurenet.co.uk
Jon Alongi CD editor
[e] jon.alongi@futurenet.co.uk

CD problems [t] 01225 822 743
Technical support: ca.support@futurenet.co.uk

Contributors
Cover illustration: Twelve:Ten
Illustrations: Kelly Boulton, Lee Hasler, Identikal, Joost Korngold, Magictorch, Kai Pedersen, Studio Output, Tado
Contributors: David Aaron, Jon'Paul Alongi, Vicki Atkinson, Michael Burns, Ryan Carson, Efrain Hernandez-Mendoza, Tim Ketterer, Nik Lever, Stephen Lockhart, Mark Penfold, Sarah Robson, Dean Utian, Garrick Webster, Kai Wood, William Yeung

Printed at: BGP

Contact
Computer Arts Projects, Future Publishing, 30 Monmouth Street, Bath BA1 2BW
[t] 01225 442 244
[f] 01225 732 295
[i] ISSN: 01225 789 293
[w] www.computerarts.co.uk

Advertising [t] 01225 442 244
Lee Haines key account manager
[e] lee.haines@futurenet.co.uk
George Lucas senior sales executive
[e] george.lucas@futurenet.co.uk
Melissa Watkins classified enquiries
[e] melissa.watkins@futurenet.co.uk
Andrew Brain advertising manager
[e] andrew.brain@futurenet.co.uk

Production
Emily Mounter production coordinator
[e] emily.mounter@futurenet.co.uk

International licensing
Simon Wear international director
[e] simon.wear@futurenet.co.uk
[t] +44 (0) 1225 442 244

Management
Kelley Corten publisher
Dom Beaven publishing director

Distributed by
Marketforce (UK) Ltd
5th Floor, Low Rise Building, Kings Reach Tower, Stamford Street, London, SE1 9LS
[t] +44 (0) 20 7633 3300

Jason Comber trade marketing manager
[e] jason.comber@futurenet.co.uk

The Future Network
Roger Parry non-executive chairman
Greg Ingham chief executive
John Bowman group finance director
Robert Price managing director

All rights reserved. All trademarks and copyrights in this issue are recognised, and are acknowledged where possible. If we have failed to credit your copyright then do please contact us – we'll happily correct any oversight. Any material submitted is accepted on the basis of a worldwide right to publish in printed or electronic form – be sure to clearly label all discs. All contents © 2003 Future Publishing Ltd.
Part of The Future Network plc, which has offices in Bath, London, Milan, New York, Paris, San Francisco and Wroclaw.

Future Publishing is part of The Future Network plc.

The Future Network produces carefully targeted specialist magazines for people who share a passion. We aim to satisfy that passion by creating titles offering value for money, reliable information, smart buying advice and which are a pleasure to read. Today we publish more than 100 magazines in the UK, US, France and Italy. Over 100 international editions of our magazines are also published in 30 other countries around the world.
The Future Network plc is a public company quoted on the London Stock Exchange (symbol: FNET). · Non-executive Chairman: Roger Parry · Chief Executive: Greg Ingham · Group Finance Director: John Bowman · Tel +44 1225 442244 · www.thefuturenetwork.plc.uk

ABC Computer Arts Projects is a member of the Audit Bureau of Circulations Jan-Dec 2003: **16,173**

COMPUTER arts projects

CONTENTS

IN-DEPTH GUIDE

TUTORIALS

14 Design a Flash portfolio

28 Create winning games in Flash

36 Build an online community

46 Flash components explained

58 Shockwave 3D in-depth

66 DVD in Director

84 Create an MP3 player

FEATURES

06 Flash and Director showcase

20 60 ActionScript tips

74 60 Lingo tips

90 The making of Virtual Replay

PROFILES

12 Electrotank

34 Float Design

54 Hi-ReS!

SUBSCRIBE

52 Free book when you subscribe!

72 North American subs offer

REGULARS

83 Back issues

96 CD contents

98 Next issue

FEATURES

06 Showcase
Check out what the best multimedia artists and developers are doing with *Flash* and *Director*...

20 60 ActionScript tips
Create things in *Flash* you never thought possible with our essential set of ActionScript tips!

74 60 Lingo tips
Master *Director*'s native programming language with these essential tips from those in the know...

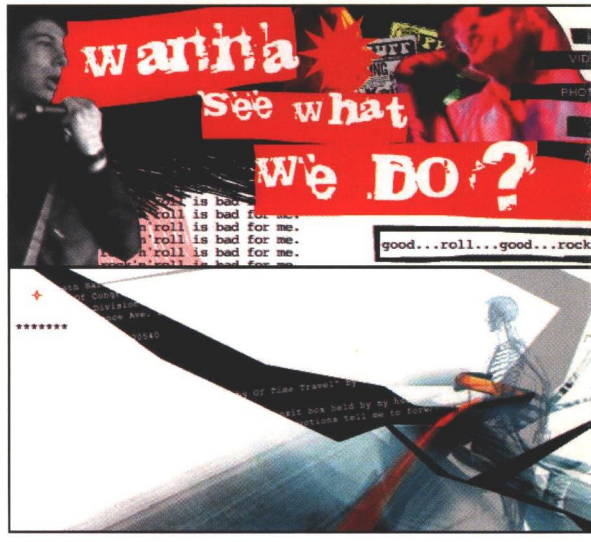
90 Virtual Replay
How the BBC recreated all the action from Euro 2004 in Shockwave 3D

PROFILES

12 Electrotank
This successful *Flash* games outfit reveals the secrets of creating stunning multimedia content for the Web

34 Float Design
It all started with a message attached to 100 balloons. Learn how Float is rising to the surface in the Web-design industry

54 Hi-ReS!
The company behind some of the most imaginative multimedia in recent times tells us where it gets its inspiration from



TUTORIALS

CREATE CUTTING-EDGE MULTIMEDIA!

TUTORIALS BEGIN ON PAGE 14...

14 Flash digital portfolios
Create a fantastic digital portfolio in *Flash* and get the dream design job you've always wanted. Expert tutorial!

28 Winning games in Flash
Build an amazing penalty shootout game using a little code and a little maths in *Flash*. All the files you need on the cover CD!

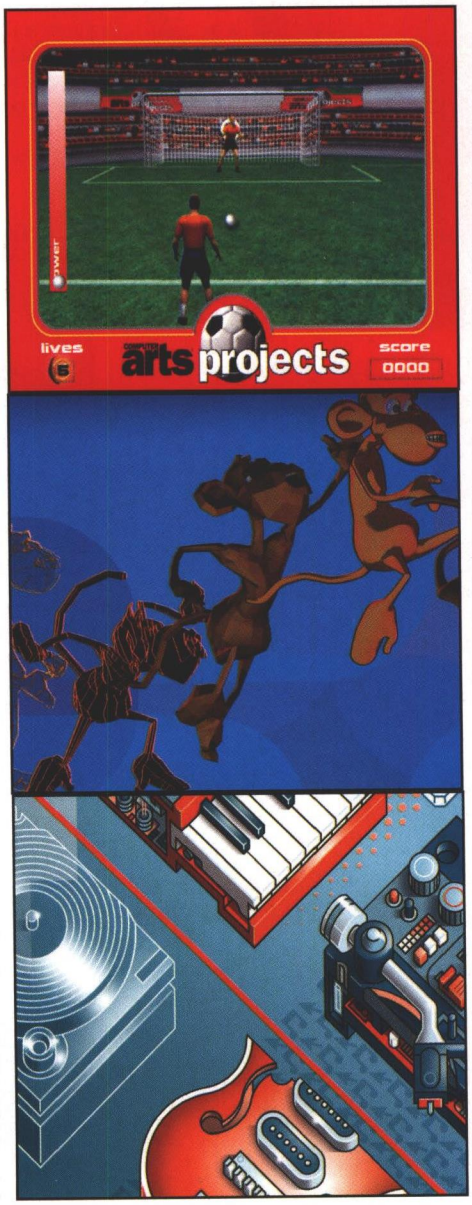
36 Build an online community
With this expert tutorial, you'll learn how to combine *Flash* and PHP to create a buzzing online community. Don't miss it!

46 Flash components
Build customisable interface elements in *Flash* using components. Every angle and aspect explained in-depth

58 Shockwave 3D
Learn the techniques you need to control a 3D character in *Director*. Essential advice!

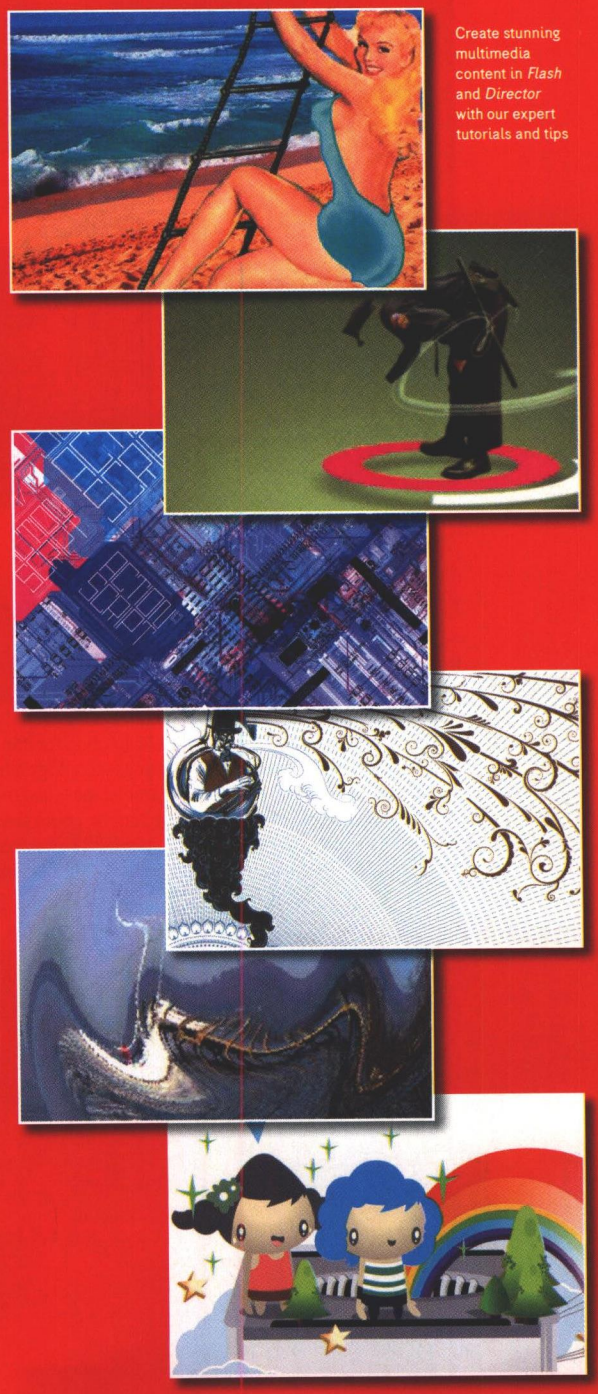
66 DVD in Director MX 2004
With *Director MX 2004* you can combine DVD content with HTML pages. Here's how to do it

84 Create an MP3 player
Create an MP3 jukebox complete with play-lists in *Director*. Expert tutorial and all the files on your cover CD



THIS ISSUE IN PICTURES...

Create stunning multimedia content in *Flash* and *Director* with our expert tutorials and tips



SHOWCASE: FLASH

See more of the best Flash content at www.macromedia.com/showcase/flash

Company > Summit Projects **Project > Nike Golf**

Nike Golf gave Summit Projects just 30 days to completely rebuild nikegolf.com. Summit turned to *Flash MX* as the most powerful solution for developing rich online content to meet all of Nike Golf's goals and get the job done on time. Working in conjunction with Nike Golf Image Design, Summit created a compelling Web presence.

Jason Herkert, Senior Designer at Nike Golf says: "Summit used *Flash MX* as a technology capable of achieving our bottom-line objectives: a cool, uncluttered look and feel that would attract a younger audience and appeal to our older, core customers, rich visuals and compelling product stories, a refined navigation and corroborative marketing capabilities. The end result is a site that effectively promotes our products and our love of the game."

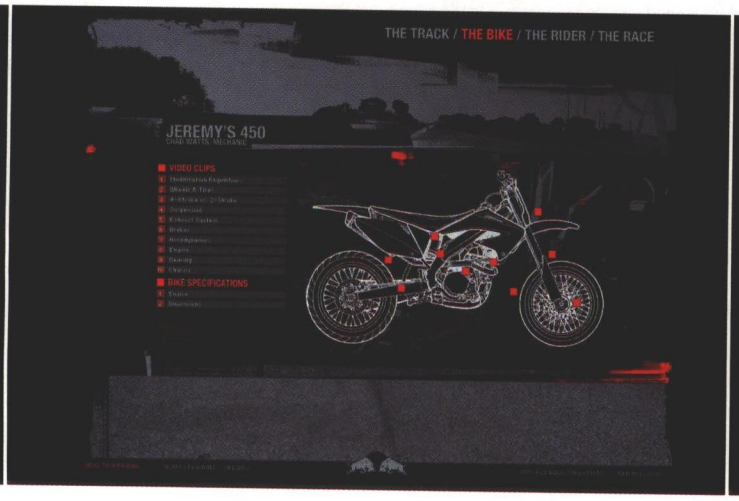
Web > www.nikegolf.com www.summitprojects.com

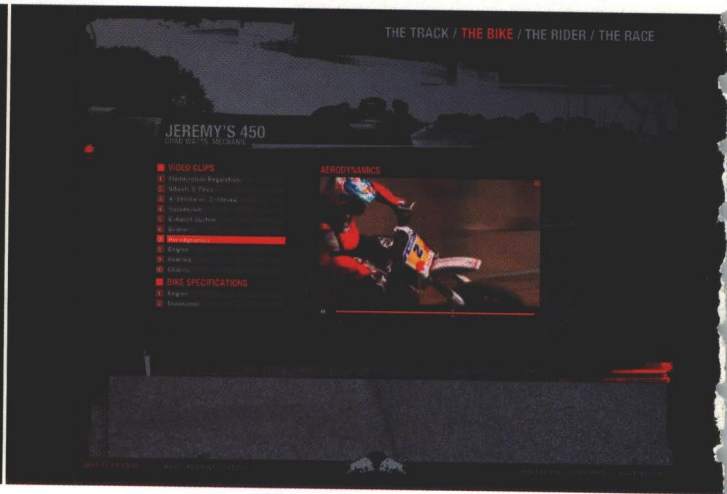
Company > Odopod **Project > Red Bull Copilot: Supermoto**

Red Bull commissioned US-based agency Odopod to transform Red Bull's video content into an interactive form. David Bliss, a co-founder of Odopod, and Technical Director of the Red Bull Copilot: Supermoto site, talks about interactive video on the Web and why Macromedia video was the only viable solution for this project:

"We never really considered using any technology other than Macromedia video, mainly because we could readily wrap logic around the video in *Flash*. Because we can put the video into *Flash*, we had great programming control, could develop our own interface elements, controls and cross-links. We could seamlessly integrate all the *Flash* content on the site with the video. There are four sections to the site and each one is 85-90 per cent video."

Web > www.odopod.com www.redbullcopilot.com





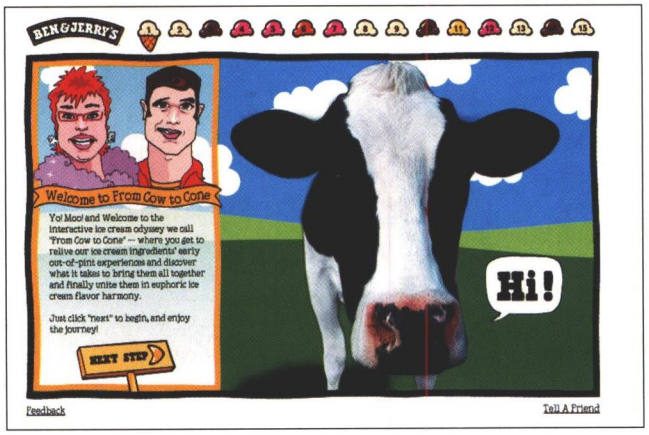
macromedia

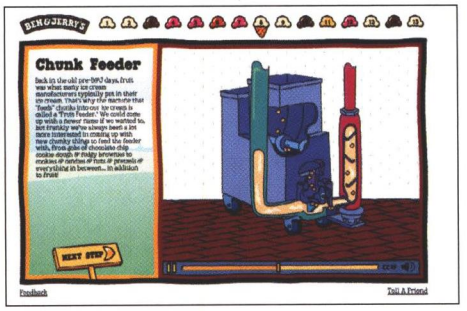
Company > Ben and Jerry's Project > From Cow to Cone

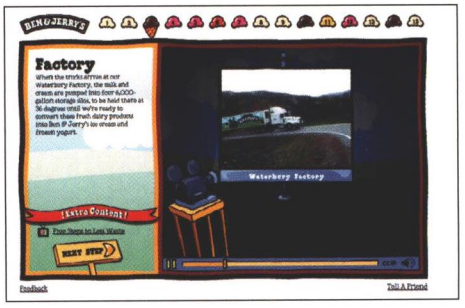
From Cow to Cone is an online application at Ben and Jerry's Website, providing the reader with a virtual tour of the ice cream giant's factory. The tour uses *Flash* to blend audio, video, and animation into a seamless user experience. *Flash* video is mixed and synchronised with text and animation. Users can navigate interactively to get behind the scenes at Ben & Jerry's for interesting insights into a company that makes headlines for the right reasons.

Bob Kilpatrick, Senior Web Developer at Ben & Jerry's, talks about embracing *Flash* video: "Without *Flash*, this project would have been out of our reach. We wanted users to sit back and be virtually transported to Vermont for an entertaining and informative experience without clicking around — not an easy thing to accomplish on the Web. Before this project, we were using *Flash* to simply animate cows. Using our existing *Flash* skills, we were able to ratchet up our site with *Flash* video to create a far richer experience than we've ever had — without spending any dollars to alter our infrastructure. As an in-house developer, I'm not able to be an expert in everything, so *Flash* and the advent of *Flash* video has been a very good way for me to grow professionally. *Flash* makes me feel like I can build anything."

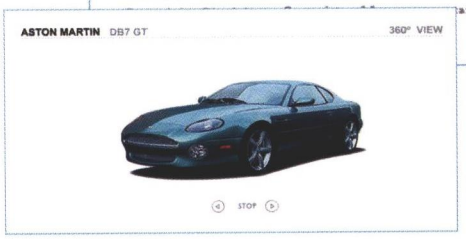
Web > www.benjerry.com


BEN & JERRY'S
Welcome to From Cow to Cone
Yo! Moo! and Welcome to the interactive ice cream odyssey we call "From Cow to Cone" -- where you get to relive our ice cream ingredients' early out-of-pint experiences and discover what it takes to bring them all together and finally unite them in euphoric ice cream flavor harmony.
Just click "next" to begin, and enjoy the journey!
NEXT STEP
Hi!
Feedback
Tell A Friend





ASTON MARTIN Vanquish
EXTERIOR
doors
Aston Martin Vanquish has two door Coupe with 2+0 or 2+2 seating and extruded aluminum door side impact protection beams.
DOORS | WHEELS | WINDOWS
FRONT VIEW SIDE VIEW REAR VIEW
SELECT A MODEL | EXTERIOR | INTERIOR | 360° VIEW | COLOR | SPECS | HELP
setup test drive
request price quote
POWERED BY izmocars



grille
Escalade's grille in-between High Intensity Discharge headlights adds to the sophisticated look of the vehicle.

Company > Izmocars Project > izmocars.com

Designing life-like animations with *Flash*, Izmocars has created virtual tours of virtually every car going. "Our goal is to capture the emotional appeal of the automobile and provide a rich online experience to prospective customers," says Tej Soni, President and CEO of Izmocars.

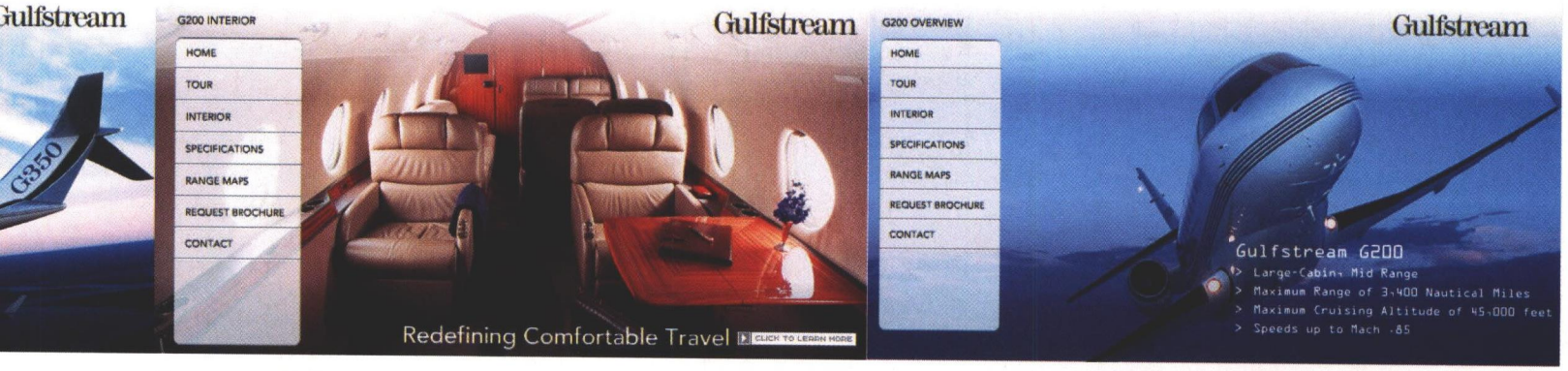
Each year, Izmocars creates animated tours of more than 400 cars generating more than 8000 *Flash* files. The animations, based on actual photographs, display realistic exterior, interior and 360-degree views of vehicles and also feature a dynamic colour selector option which changes the colour of the vehicle in real time with the click of the mouse.

According to Soni, by using *Flash*, Izmocars can develop animated car tours faster and at a lower cost than using any other technology. Plus, the experience is more seamless and visually engaging. "With *Flash*, we can develop these projects in 30 per cent less time than it would take to develop content using any other product. And thanks to vector technology and the *Flash* Player, the experience loads quickly and seamlessly."

Web > www.izmocars.com

SHOWCASE: DIRECTOR

See more of the best Director content at www.macromedia.com/showcase/director



Company > Gulfstream Aerospace **Project > Gulfstream Aerospace DVD**

Private Jet company Gulfstream Aerospace has manufactured more than 1400 aircraft for corporate, government, private and military customers around the world. When its sales force needed a new, engaging and easy-to-navigate marketing and sales tool, they wanted more than just a leave-behind pamphlet. What was needed was a combination of video and other interactive assets, something that raised the bar of sophistication when the sales executive was meeting with a client.

"The purpose of it is really to provide a direct marketing tool and a sales tool, so that the sales force can sit with a client in front of a high-quality video and navigate through these different parts of the video in an immersive presentation," says Will Dent, Manager of interactive marketing for Gulfstream Aerospace.

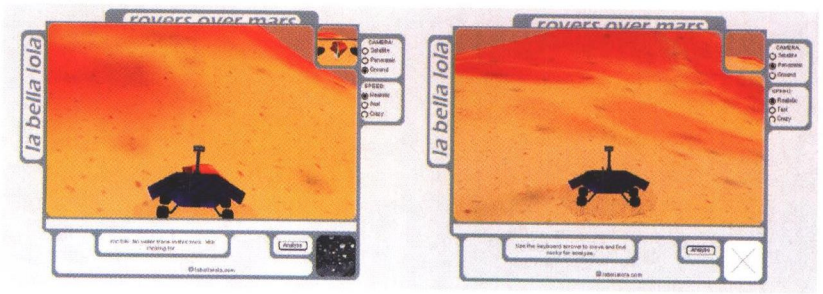
He continues: "This version of *Director* completes the MX 2004 product line, as far as development and integration. Because I know we're going to do more DVD/CD-ROM Website projects, what this means is truly being able to lever and repurpose all my materials including code, components and .swc files, which greatly improves our workflow."

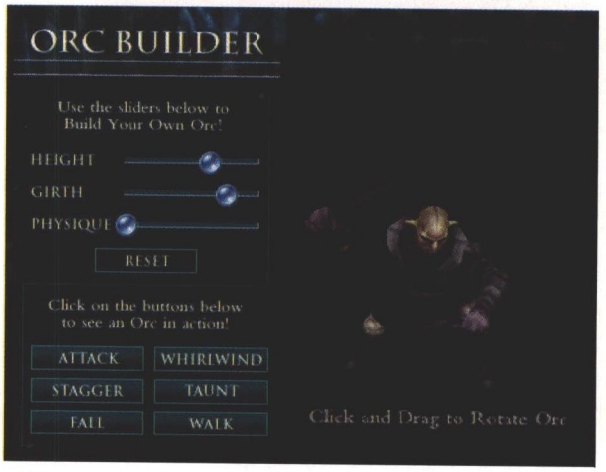
Web > www.gulfstream.com

Company > La Bella Lola **Project > Rover Over Mars**

La Bella Lola uses Macromedia *Director MX* to deliver an interactive game that enables users to drive a NASA Rover around Mars searching for traces of water. The result is a realistic Shockwave 3D game.

Web > www.labellalola.com/mars/



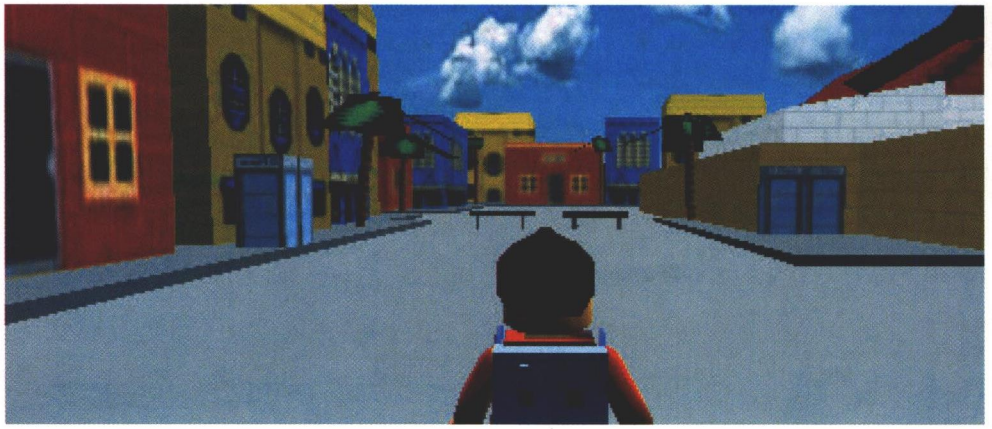


Company > Giant Wheel Production
Project > The Lord Of The Rings: Build Your Own Orc

The official *The Lord of the Rings* Website contains unique companion modules that feature exclusive behind-the-scenes content. The 'Inside the Effects' module features an interactive 3D Build Your Own Orc application that simulates the results produced by the Orc Builder application used for the actual movie production. The application also simulates the motions built for the characters by the special effects experts.

The application was built in *Director* by Giant Wheel Productions and uses Shockwave 3D extensively.

Web > www.lordoftherings.net www.giantwheelproductions.com

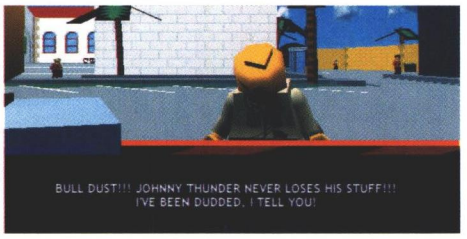


Company > Templar Studios LLC **Project >** LEGO Studios Backlot Game

The LEGO Company, based in Billund, Denmark, is a world leader in providing quality products and experiences that stimulate children's creativity, imagination and learning. The LEGO Studios Backlot game introduces kids and parents to the LEGO Studio product line through an interactive gaming experience. Players explore a simulated movie studio where they can get a job, learn about filmmaking, and find clues to help solve a mystery.

The game was built by Templar Studios LLC using *Director* and integrating Shockwave 3D technology. The brief was to create a high-quality 3D game that has console quality game-play that could be delivered via the Web.

Web > www.lego.com/studios/backlot/default.asp www.templar.com



Company > Skyworks Technology **Project >** Candystand.com

This Kraft Foods site attracts more than 20 million visitors each month. Why? Because of the outstanding *Director*-based games. You'll find everything from American football to mini golf, all with a sweet or biscuit-based theme. Some truly addictive games here.

Web > www.skyworks.com www.candystand.com

CONTENTS

SECTION ONE: CUTTING-EDGE FLASH

Flash has become the industry standard tool for creating fantastic multimedia, games and Web applications. Over the following section, you'll learn to do things in this app you never thought possible!

Over the last few years, Flash design has stepped up a gear. A tool that was once widely used for splash screen animations has become a full-blown Web development app, capable of creating wonderfully-rich multimedia content. Over the next 40 pages, we bring you some brand new tutorials – you'll learn how to create an exciting penalty shootout game, build a dynamic portfolio, combine Flash and PHP to create an online community and more! Read on for the highlights...

p12 _

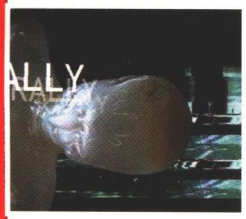
PROFILE: Electrotank
The company responsible for the incredibly addictive Mini Golf *Flash* game reveals the secrets behind creating amazing sticky content.

p36 _

TUTORIAL: Build an online community
In this exclusive tutorial from Float Design (see profile on page 34) you'll learn how to combine *Flash* and PHP to create an online community.

p46 _

TUTORIAL: Flash Components
In this expert tutorial we bring you all you need to know about creating customisable components in *Flash MX 2004*.

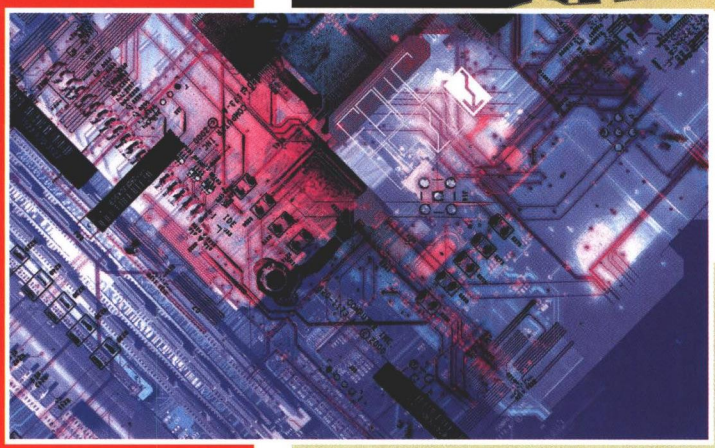
p54 _ 

PROFILE: Hi-ReS!
This London-based outfit is one of the biggest players in multimedia content creation, with a client list including MTV, HBO and Massive Attack!

TUTORIAL: Flash digital portfolio

If you're struggling to show off your work in style, follow this expert tutorial and create a visually-striking digital portfolio in *Flash*. You'll find all the files you need to complete the tutorial on your cover CD.

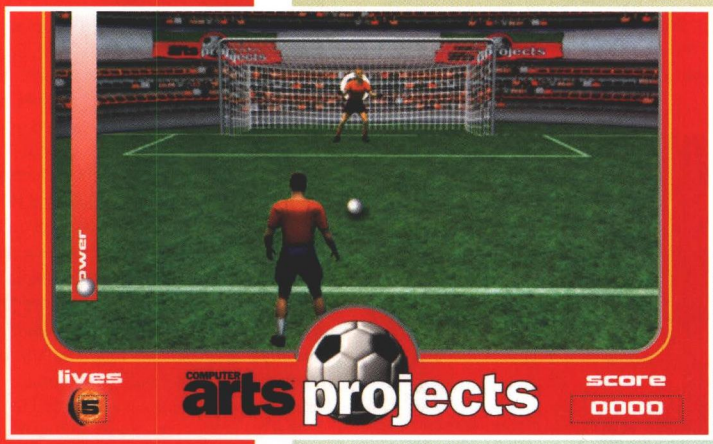
p14 _



FEATURE: 60 ActionScript tips

Learn how to do things in *Flash* that you never thought possible with these 60 ActionScript tips. Whether you're new to the programming language or a seasoned expert, there'll be something here for you!

p20_



TUTORIAL: Winning games in Flash

Forget about Beckham missing that penalty and create an awesome penalty shootout game of your own. Learn the basic maths and ActionScript involved in order to curl the ball into the back of the net...

p28 _

Electrotank

"Our number one goal is to create great games." Electrotank's Robert Firebaugh shares the secrets of successful *Flash* games...

For a studio's first production to experience the success of Electrotank's *Mini Golf* is no easy task. The simple but addictive online game was an instant hit with Web users the world over for its great gameplay and entertaining use of *Flash*. But that was three years ago. Since then the small game development outfit has expanded its portfolio to over 15 strategy, puzzle or multiplayer games, won several awards and written three books on *Flash* game development.

Think-tank

Electrotank was founded by three game addicts – Jobe Makar, President; Mike Grundvig, Chief Technical Officer; and Robert Firebaugh, Creative Director. Based in North Carolina in the US, the outfit has a small group of designers, and drafts in freelancers for sound production and bigger projects. For Robert Firebaugh, Electrotank has been a great way to fuse a successful career in new media with a love of games. Founding the studio with Jobe and Mike meant combining his own illustration and animation talents with *Flash's* powerful graphics tools. "When we started, it was just another hobby, a way to channel my creative juices because my day job at the time was not very challenging. I didn't have any aims other than having fun and creating great games."

Great looking with great sound effects, *Air Hockey* will leave you wondering whether it really was created just in *Flash*. But it was.

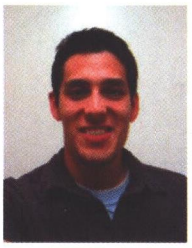
Not just a free game on a PC, *Solitaire* has been taken to new heights with a collection of card-based games in Electrotank's Regal Solitaire series. "Surprisingly, there is a lot of strategy that goes into playing *Solitaire*," says Robert.

Over three years later, the trio's aims are still the same, with the added challenge of finding financial stability in an unpredictable industry. It's risen above the problems faced by other studios by offering game design or existing licences to clients as well as giving leisure gamers the chance to upgrade from free demos of their games to full versions for a nominal fee.

Multiplayer gaming

A recent addition to the business plan is a multiplayer server product developed by Mike Grundvig. ElectroServer is a piece of software that links an unlimited number of clients together. All of Electrotank's own games are based on this server technology and now they are offering it to clients. To date, the proprietary software has attracted clients such as Nickelodeon, MiniClip and Rubik. "ElectroServer makes chatting and multiplayer games possible by intelligently exchanging data between clients at super speeds," Robert explains. "The majority of socket-servers out there have a problem with stability at high numbers of users. ElectroServer has been tested to over 5,000 simultaneous users with no stability issues." A 20-user version is currently available as a free download from Electrotank's site.

But it's not just the backend that makes Electrotank one of the hottest names in game development. The range of games, styles and sounds in its portfolio



electrotank

Co-founder and Creative Director, Robert Firebaugh

continues to attract gamers to its Website. "We don't like to stick with one style – the type of game dictates how each game will look, sound and feel," says Robert. "The projects we enjoy working on most are those with a distinct artistic style or vibe, because you have better odds of creating a game that no one has ever seen, making it more appealing."

So what does make a good game? "That's like asking the secret of the universe!" he exclaims. "A good game is one a player will enjoy. The only way you will know if a player will enjoy it is to put yourself in their shoes. Understand who is going to play your games and design it around pleasing them. People play games to have fun so don't set them up to lose."

A sneak peak at a brand new project from Electrotank towers. *Tankcore* is a real-time multiplayer warfare game. In Robert's words, "It'll be a real jaw dropper!"

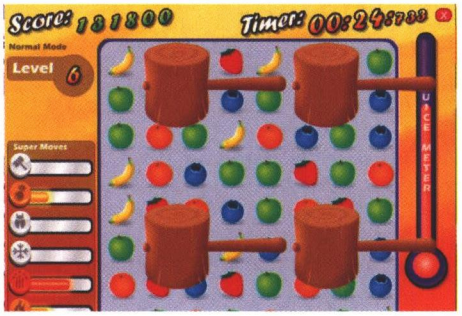
Thorough planning

Electrotank takes its planning for each game seriously, with no development work until the idea and scope of the game is thoroughly drafted out. Then time, budget or technical issues are taken into account before building a test engine to see how it will work. It's only when this has been approved that the art and animation work begins, followed by the sound effects. Finally, it gets tested thoroughly. "Get feedback from as many people as possible," recommends Robert. "After working on a project for a while, little things can slip by that someone with a fresh look will spot."

All its games are designed in *Flash* and use ActionScript to connect to the server. It's not just the latest version either, the art team uses a combination of *Flash MX 2004* and *Flash 4* due to some features in older versions that help with art production. But for development, it's MX all the way. "With newer versions of *Flash*, the possibilities are much greater than when we first started," says Robert. "The online gaming industry has grown, there is definitely a lot more competition now."

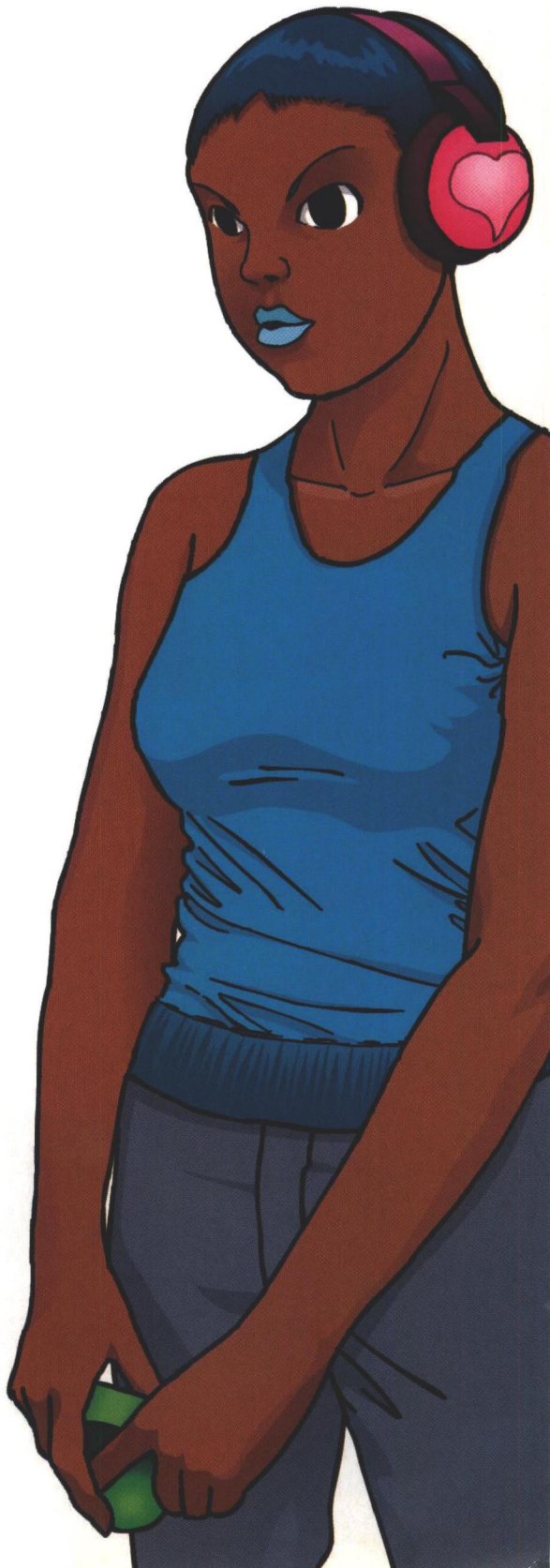
But as far as he's concerned, the industry is still in its beginnings. "The way the user interacts with a mouse and keyboard through a browser is still relatively new," says Robert. "I feel we still have a way to go before massive improvement is made in online game design." ca-p

Mini Golf was Electrotank's first online game production and was a massive hit. See how the simple vector shapes compare with the more advanced characters of *Air Hockey*.



Fruit Smash is a take on the traditional stacking games. Move fruits into lines to make them disappear and beat the clock. With great use of *Flash*, it's colourful and highly playable.

You can contact Electrotank via its Website at **[w]** www.electrotank.com or call **[t]** 001 919 609 0408. For more information on its ElectroServer software and to download the free version, see **[w]** www.electro-server.com.



Flash digital portfolios

In this tutorial, we're going to show you how to use *Flash* to create a simple, yet elegant, digital portfolio for distribution on CD...

When applying for jobs these days, it's essential to have a digital portfolio of your work. And one of the best ways to do this is by placing your portfolio on a CD and making some lovely packaging to hold it. Creating a physical product to hold your digital portfolio will give you an advantage over people who have only created Web-based portfolios. This is because receiving a package through the post is much more attention-grabbing than simply sending an email – it really shows you've made an effort!

With this in mind, we're going to show you how to build a digital portfolio in *Flash*, the primary goal being to get you noticed by potential employers. We'll then show you how to export it to run on a CD. All you'll need is a copy of *Flash MX*, some blank CDs, a CD burner and some patience.

Expertise provided by Ryan Carson. You can contact Ryan at
[e] ryan@carsonsystems.com or visit
[w] www.carsonsystems.com.

All the components needed to complete this tutorial can be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Flash portfolio folder.

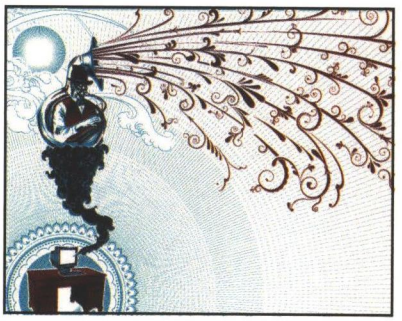
Illustration by Magictorch

Part 1 : Creating the interface

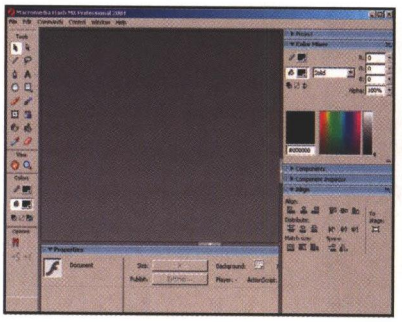
Build the interface to your portfolio using one of your existing illustrations...

Make it simple

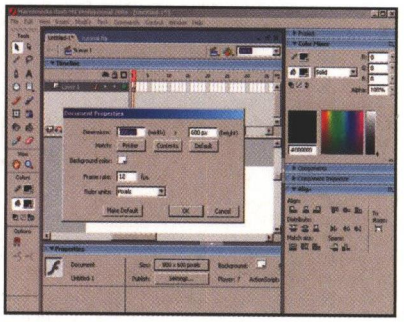
When creating your digital portfolio, it's important to make it easy to use. If a potential employer can't figure out how to navigate and use your portfolio, they're not going to employ you. The first thing to do is make sure that it loads quickly. Secondly, make it immediately obvious how to use the portfolio menu. This isn't the place to experiment.



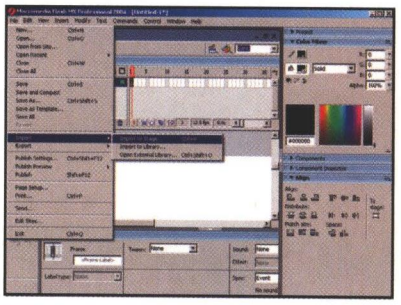
1 First, you need to lay out the general interface of the portfolio. We're going to use an illustration by Magictorch (**[w]** www.magictorch.com) as the background. Make sure your interface design has a clear area for a menu.



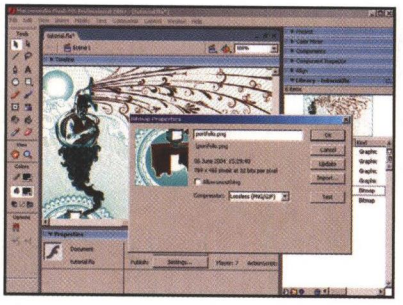
2 Next, open *Flash*. We're using *Flash MX 2004 Pro*, but you can use *MX* or *MX 2004*. If you don't have *Flash*, you can download a free trial version from **[w]** www.macromedia.com/flash.



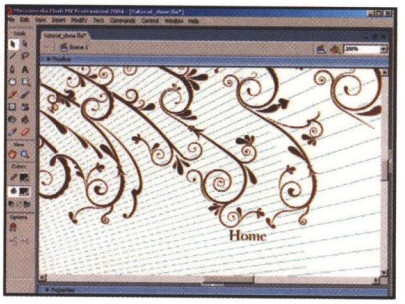
3 Create a new document (File>New>Flash Document) and set the dimensions to 800 pixels wide by 600 pixels high by going to Modify>Document. We left the default frames per second at 12, but you can adjust this value.



4 Now we're going to import our image. To import, choose File>Import>Import to Stage. Browse for the image you've chosen to use, and hit Open. You could also create the interface graphics right inside *Flash*, using its drawing tools.



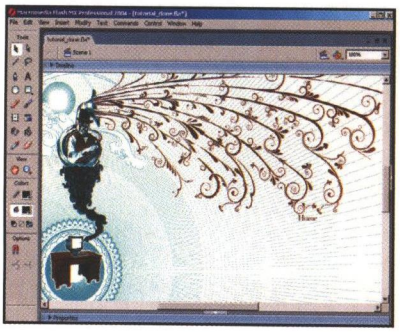
5 *Flash* has a rather annoying tendency to blur images that have been imported. To avoid this, open the Library (Window>Library) and double-click the image you imported. Uncheck Allow smoothing, and hit OK.



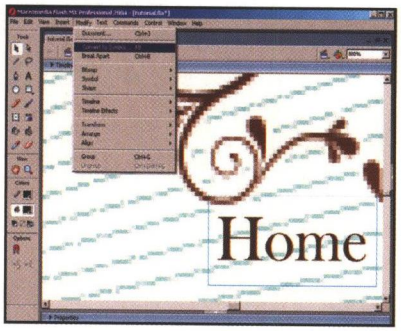
6 Now it's time to create the menu. We're going to have three links: Home, About and Work. These are fairly typical for portfolios. You might also want to add a Contact link, if you don't include your details in the About section.

Part 2: Navigation animation

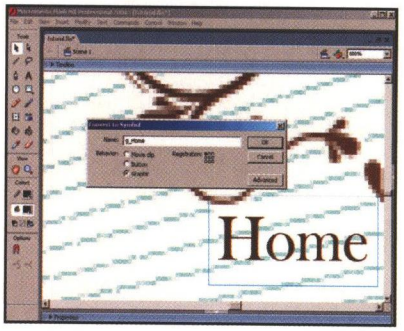
Capture the viewer's attention with a simple animation...



1 It's important that the user can see the navigation easily when the page first loads. As ours is tucked away a bit, we need to draw the user's eye to it when the CD starts.



2 We're going to do this with a simple animation – the navigation words will slide onto the page from the borders. To do this, start by selecting Home and choose Modify>Convert to Symbol from the menu.



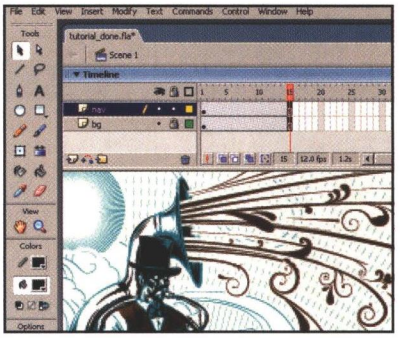
3 Enter b_Home for the name, select the Button radio button and hit OK. We've added the b_ prefix to the name to make it easy to identify as a Button Symbol.

Use the space

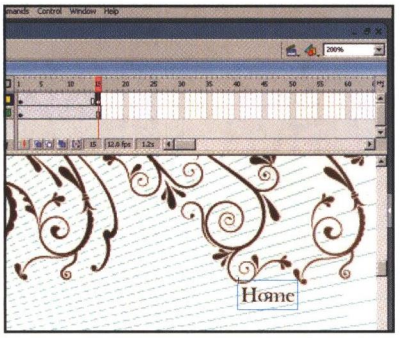
When building a portfolio that will be placed on CD-ROM, you can afford to use larger images and media. This is something you should take advantage of in order to really impress your viewer. However, don't go overboard on this one. Many viewers have slower computers that croak when viewing huge media files.

Spread the word

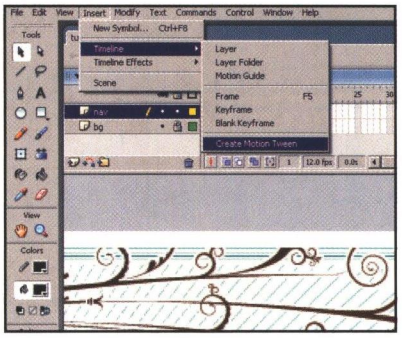
Be crafty! If you're looking to get the word out about your portfolio, be creative about distributing it. Think about who your viewers are and where they hang out – are there any opportunities to put your portfolio in their path? A good idea is to do a gallery or exhibit and have plenty of your portfolio CDs ready to hand out to visitors.



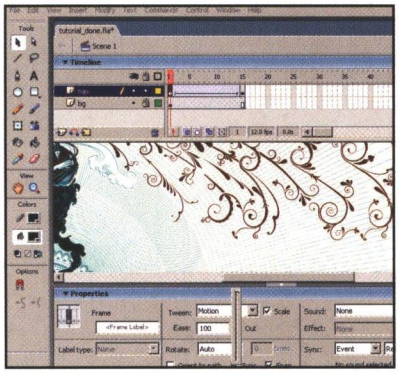
4 Add a new layer, called nav, by going to Insert>Timeline>Layer, and cut and paste b_Home onto it. Select frame 15 on the Timeline by going to Window>Timeline, and hit F5 to insert a new frame. Repeat this for the other layer (bg, in our example).



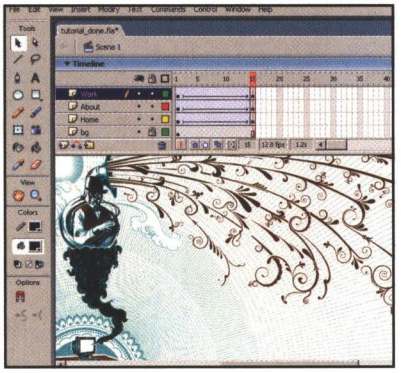
5 Next, add a keyframe on frame 15 on the nav layer. Select frame 15 and choose Insert>Timeline>Keyframe from the menu. This is going to enable you to move the b_Home symbol onto the page.



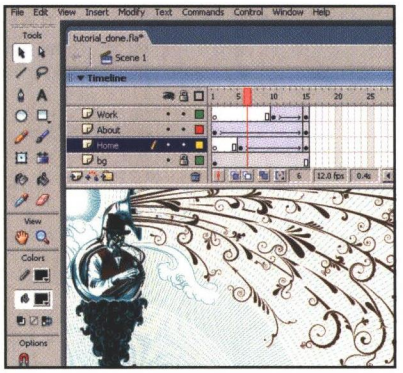
6 On frame 1, select the b_Home symbol that's already been created and move it off the canvas. Now select frame 1 and choose Insert>Timeline>Create Motion Tween.



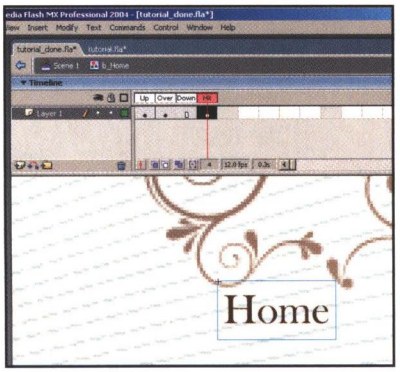
7 A great effect to add to motion tweens is Easing. This makes the motion tween look a bit more elegant. To add Easing, select frame 1 on the Properties panel (Window>Properties) and change Ease to 100 (Out).



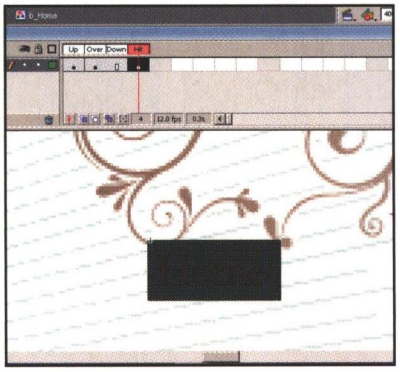
8 Now that you've added the Home nav element, go ahead and repeat steps 3-7 with About and Work. You should end up with four layers in total. Experiment with the motion tween Easing – it's quite versatile.



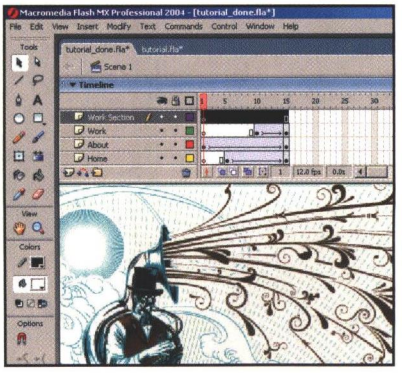
9 We've decided to stagger the beginning frames of the nav elements, because it makes the animation more interesting. Feel free to play around with this. Just make sure the animation doesn't last any longer than one second – to prevent your user from becoming impatient.



10 You now need to modify the three nav buttons. Double-click on b_Home to enter its Timeline. Select the Hit frame and hit F6. Do the same with the Over frame. This creates keyframes.



11 On the Hit frame, add a solid box over the text. This determines where the hot-spot for the button is. You can change the Over state to another colour to add to the usability. Follow this and the previous step for the other nav buttons.



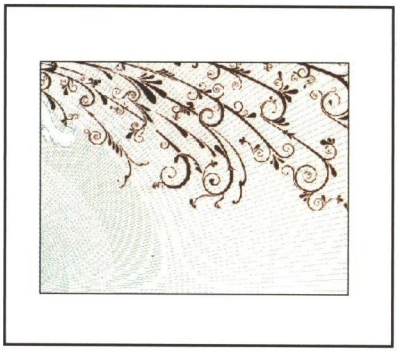
12 Now that you've created the buttons on the interface, you need to create the most important part of the portfolio: the Work section. To do this, add another layer, called Work Section, to the Timeline.

Part 3: The browser

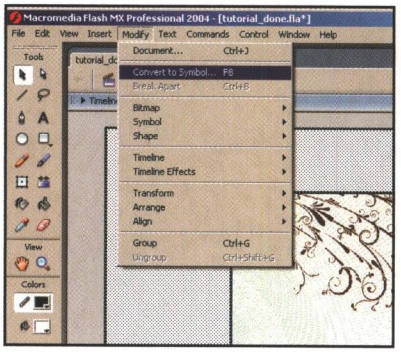
Everything is set up and ready to go, so now it's time to add an image browser...

Create the right impression

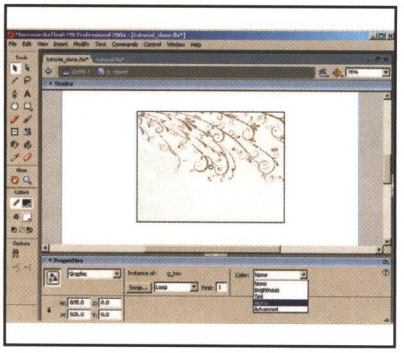
A sure-fire way to get your portfolio noticed is by placing it in some really funky packaging. If you can get that creative director or company owner to notice your CD, you've won half the battle. If it looks boring on the outside, they'll assume the content of the CD must be boring as well.



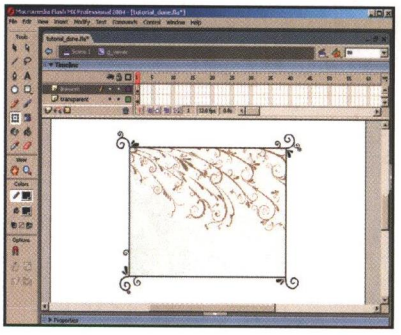
1 On the Work Section layer, draw a white rectangle that covers the whole interface (805 high x 605 wide) and then draw another box inside it (400 high x 300 wide). Make sure the boxes are centred on each other.



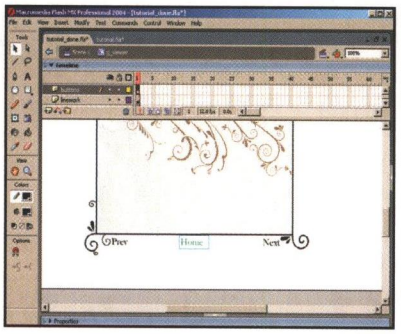
2 This is going to be a simple image browser that enables the user to see past work. Now select the inner box fill and delete it. Then select the outer box and choose Modify>Convert to Symbol from the menu. Type m_browser, choose Movie clip and hit OK.



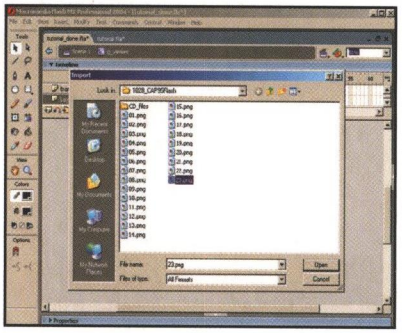
3 Double-click this new movie to enter its timeframe. Select the outer box (but not the inner box outline) and convert it to a Graphic Symbol (Modify>Convert to Symbol). Select it and open up the Properties panel. Choose Alpha from the Colour dropdown menu and enter 80 per cent.



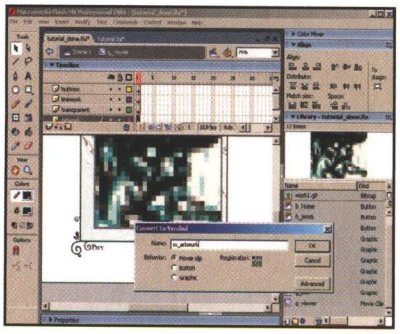
4 Place the inner box outline on a layer on top of the graphic that you just made transparent. Go ahead and add some decoration to this. We've made the linework match the look and feel of the main interface.



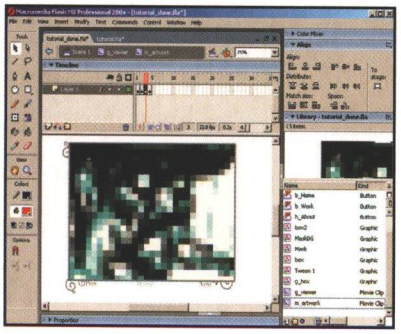
5 Now you need to add the Next, Previous and Home buttons. This enables the user to return to the main interface when they've finished browsing your images. Type the text and convert them into Button Symbols.



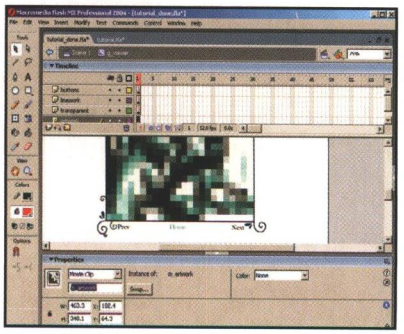
6 Now repeat Steps 10 and 11 in Part 2 to modify the buttons and make them useable. Now that you've got the buttons set up, you need to add the images to browse. Create a new layer called Artwork and import them into *Flash* (File>Import>Import to Stage).



7 Select your imported images and choose Modify>Convert to Symbol. Type m_artwork, choose Movie clip and hit OK. Double-click on the new m_artwork movie.



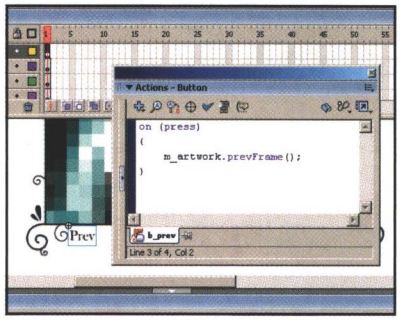
8 Now, simply create a keyframe for each image and place one image on each keyframe. By clicking Next or Previous, the user is simply going to proceed forwards or backwards through this movie – one frame at a time. On each of these frames, add a "stop();" action.



9 Select the m_artwork movie and open the Properties panel. In the <instance name> field, type m_artwork. Now we'll add the simple ActionScript that controls the m_artwork movie, based on which button the user clicks.

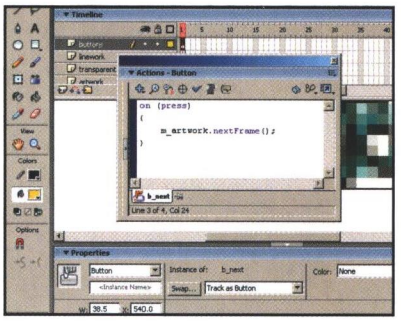
Get straight into it

Avoid any sort of intro page. These screens make it harder for users to actually see what they came for – your work – so avoid them like the plague. It's always important to remember that while looking good, digital portfolios should always be designed for people who are in a hurry. You don't want them to close it down before you've really had a chance to show them how creative you are.



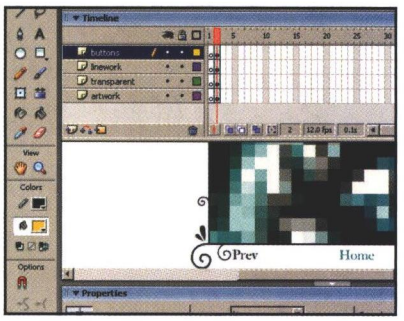
10 Select the Previous button and hit F9 to open the ActionScript panel. Type:

```
on (press)
{
m_artwork.prevFrame();
}
```

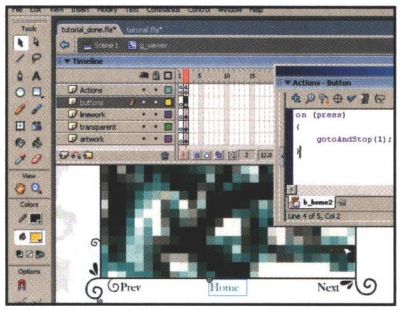


11 Now select the Next button and enter the following ActionScript:

```
on (press)
{
m_artwork.nextFrame();
}
```

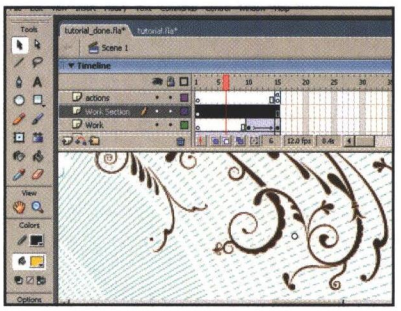


12 The user can now browse through your images. You need to allow them to close the image viewer if they want to return to the main interface. Select all four layers and drag them one frame to the right. This will create a blank keyframe on frame 1.

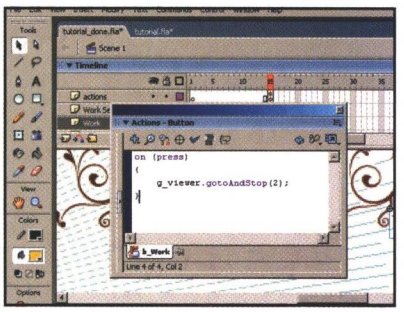


13 Add a layer called Actions and add the action "stop();" to both frame 1 and 2. Now select the Home button and type:

```
on (press)
{
gotoAndStop(1);
}
```



14 Name the image browser you created so it can be controlled by ActionScript. Select the layer it's on, Work Section, and you'll see a white circle appear. Click on that and in the Properties panel, give it the name g_viewer.

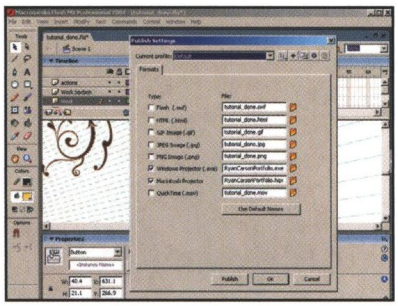


15 On the main timeline, the image browser has now disappeared. You need to add ActionScript to the Work button to make it appear:

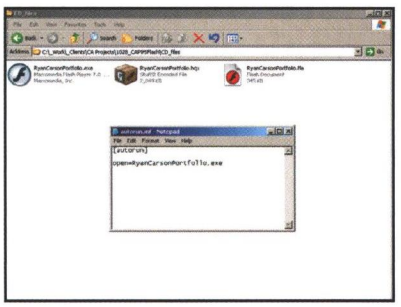
```
on (press)
{
g_viewer.gotoAndStop(2);
}
```

Part 4: Publishing your portfolio

Now you have your portfolio, here's how to get it onto a CD or DVD...

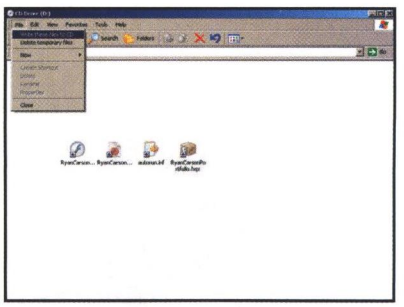


1 It's time to show you how to export your portfolio and burn it to CD. Choose File>Publish Settings from the menu. Tick the Windows Projector and Macintosh Projector boxes. Then hit Publish.

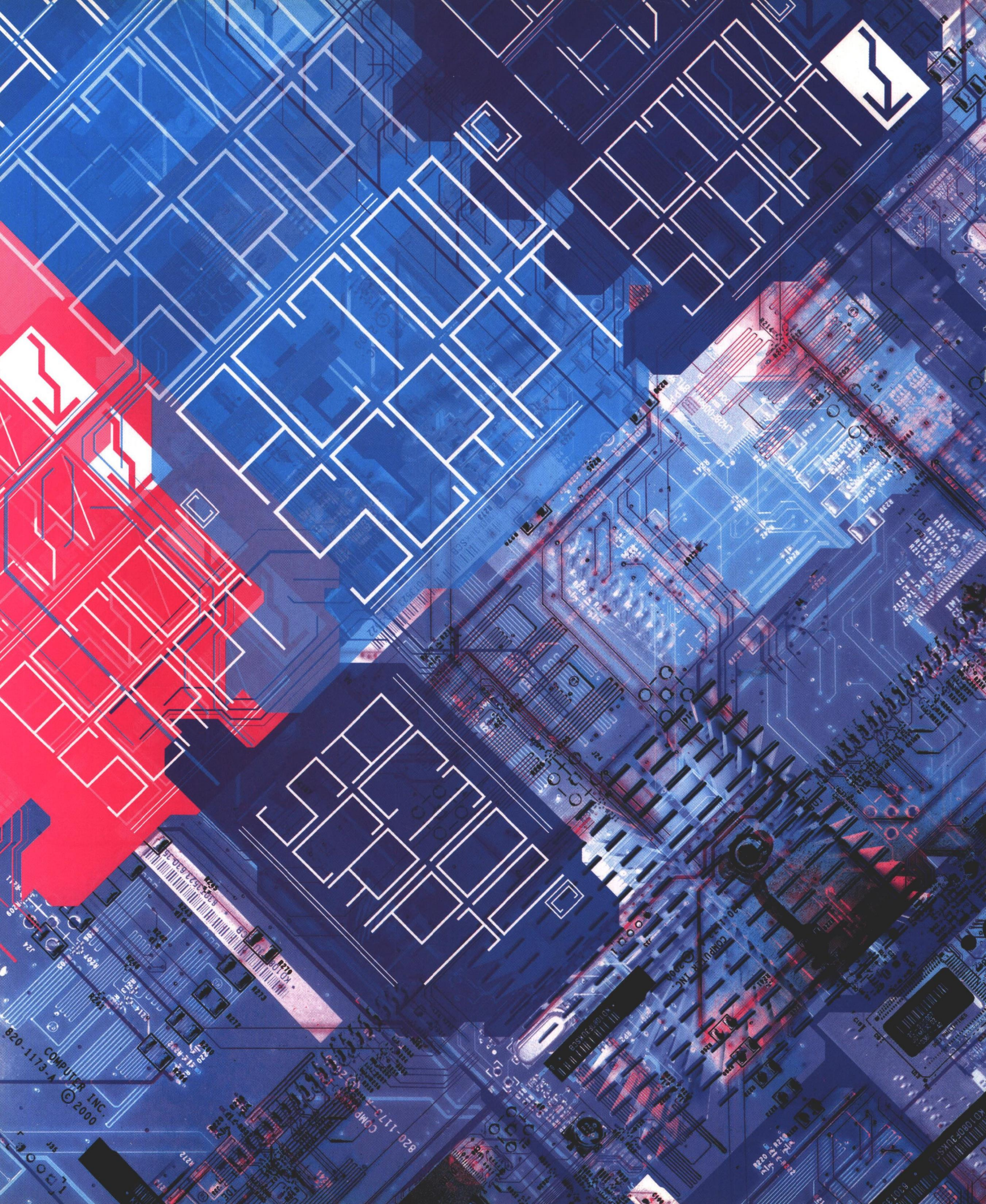


2 If you're on a PC, you can make the disc start as soon as it's put in the CD drive. OS X doesn't allow this. Create a text file called autorun.inf with this text:

```
[autorun]
open=RyanCarsonPortfolio.exe
```



3 Place a blank CD-R into your drive, drag these files onto the CD Drive icon and choose File>Write these files to CD. You're now on your way to creating a portfolio CD that will really grab people's attention! Good luck. ca-p


Beginning
Shell
Scripting

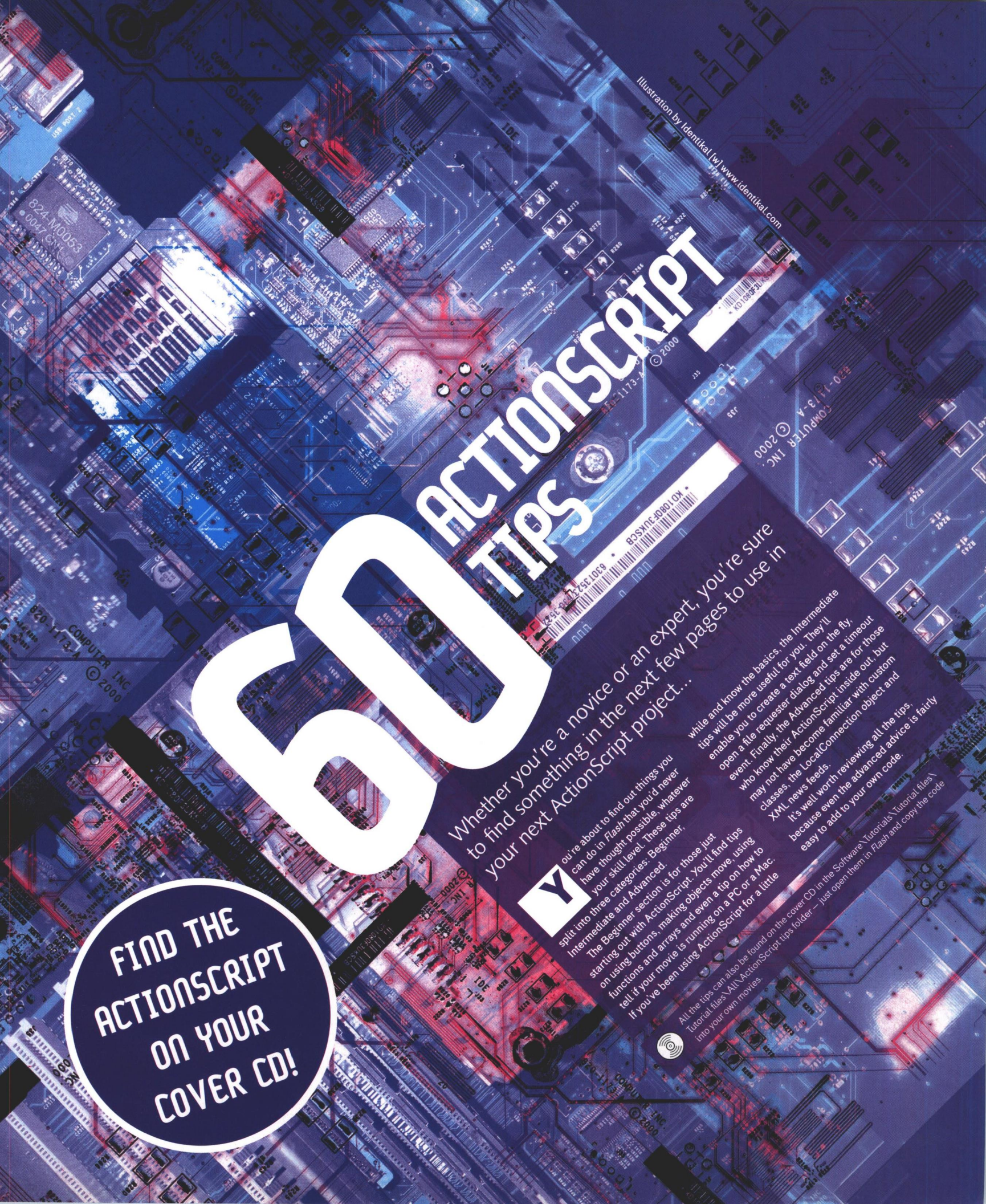


Illustration by Identikal [w] www.identikal.com

60 ACTIONSCRIPT TIPS

Whether you're a novice or an expert, you're sure to find something in the next few pages to use in your next ActionScript project...

You're about to find out things you can do in *Flash* that you'd never have thought possible, whatever your skill level. These tips are split into three categories: Beginner, Intermediate and Advanced.

The Beginner section is for those just starting out with ActionScript. You'll find tips on using buttons, making objects move using functions and arrays and even a tip on how to tell if your movie is running on a PC or a Mac.

If you've been using ActionScript for a little while and know the basics, the Intermediate tips will be more useful for you. They'll enable you to create a text field on the fly, open a file requester dialog and set a timeout event. Finally, the Advanced tips are for those who know their ActionScript inside out, but may not have become familiar with custom classes, the LocalConnection object and XML news feeds.

It's well worth reviewing all the tips, because even the advanced advice is fairly easy to add to your own code.

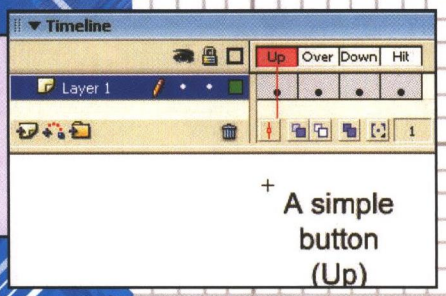
All the tips can also be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\ActionScript tips folder – just open them in *Flash* and copy the code into your own movies.

FIND THE ACTIONSCRIPT ON YOUR COVER CD!

60 ACTIONSCRIPT TIPS

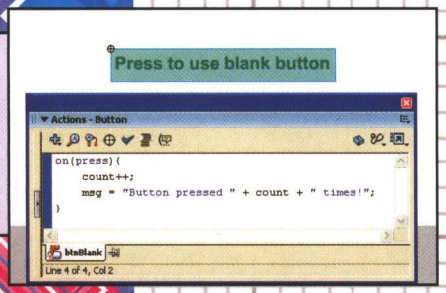
TIP 1

You can create a button using a MovieClip comprising four frames.



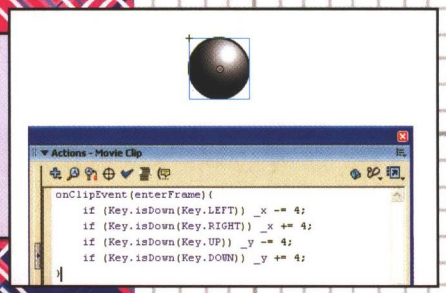
TIP 2

A blank button is the simplest type to create, using just one frame.



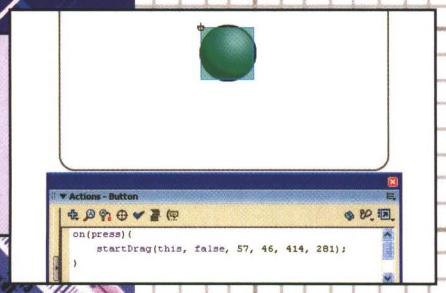
TIP 5

Add this simple code to move objects with the arrow keys.



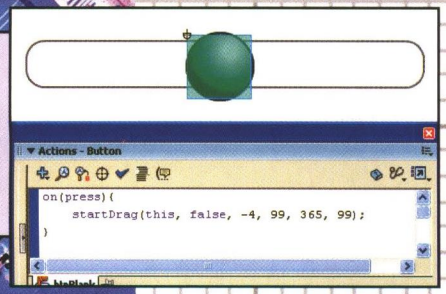
TIP 7

Limit the drag region of an object by adding values to the call to startDrag.



TIP 8

You can create a simple scroll bar by limiting the y values.



1. Creating a button

Buttons are just a special kind of four-frame MovieClip. The four frames are the up, over and down images and the hit area, in that order. You can add a script to these frames by clicking the button and adding events for press, release, rollover and rollout.

2. Using a blank button

The simplest kind of button you can create contains only the fourth frame: the hit area. A blank button is displayed in turquoise at design time, but is blank at runtime. You can simply place them over the top of other objects to create a clickable button.

3. A multistate button

Sometimes you'll need a button that can be set to multiple states. To do this, create a button, select it, then press F8, and create a MovieClip symbol from the button. Now when the button is clicked, you'll be taken to a new frame inside the clip.

4. Jumping with a button

Often a button will be used to navigate a movie. Simply add the following code:

```
on(press){
 gotoAndStop("MyMarker");
}
```

This causes the playback of the timeline where the button is placed to jump to the frame labelled MyMarker.

5. Moving with the arrow keys

To move an object on screen it should be a MovieClip. You can add code to the MovieClip by adding an onClipEvent handler. In this example, we add an onClipEvent(enterFrame) handler. We check the keyboard using the Key object. If the arrow keys are pressed, we alter the position of the MovieClip by adjusting the _x and _y values.

6. Moving by dragging

Create a MovieClip with a blank button inside. Add the following code to the button and you'll have a draggable clip:

```
on(press){
 startDrag(this, false);
}

on(release){
 stopDrag();
}
```

7. Limiting the drag region

Often you'll want to limit the region in which the dragged object can move. This is done by adding four additional parameters to the call to startDrag: left, top, right and bottom values. To find these values, use the following code to display the current position of your clip:

```
trace(_x + ", " + _y);
```

Write down the values at the top-left and bottom-right corners and use these in the call to startDrag.

8. Simple scroll bar

You can use the draggable clip to create a simple scroll bar. Just limit the y values to the same value for top and bottom and you have a horizontal scroll bar. You can use the _x value to give a value for the scroll bar using the following code, where left and right are the values used in startDrag, and limit is the maximum value:

```
value = ((_x - left)/(right -
left)) * limit;
```

9. Link the scroll bar to a text box

To display the value, create a text box. Use the Properties window to set it to Dynamic and enter a variable name, in this case 'scroll.value'. 'scroll' is the instance name for the scroll bar. The value 'value' is set when the button is released.

10. Moving in a circle

To move in a circle, use the Math object methods sin and cos. If the radius of the circle is x, then the position of the object at angle theta is:

```
_x = x * Math.cos(theta);
_y = x * Math.sin(theta);
```

To centre this in the Movie, add half the width to the _x value and half the height to the _y value.

11. Creating a function

Functions are very useful for keeping a block of code in one place. Use the keyword function followed by your choice of name and then brackets. The code for the function is contained between curly braces.

```
function myfunction(){
 trace("Called myfunction");
}
```

You call the function simply by using the name of the function and brackets.

```
myfunction();
```

You can pass parameters to a function using this code:

```
myfunction("Hello world");

function myfunction(str){
  trace(str);
}
```

12. Returning a value from a function

To return a value from a function, use the return keyword. In this example, the output from trace will be 16.

```
trace (sum(11, 5));

function sum(a, b){
  return (a + b);
}
```

13. Creating an Array

Arrays are useful for storing your information. You can create an Array using the new keyword:

```
var emptyArray = new Array()
var populatedArray = new
Array(1, 2, 3, 4, "a", "b",
false);
```

14. Accessing Array elements

```
var myArray = new
Array("Macromedia", "Flash",
"MX", "2004");
```

To access a member of myArray use myArray[i], where 'i' can be any value from 0 to one less than the number of elements in the Array.

```
myArray[0] is "Macromedia"
myArray[1] is "Flash" etc.
The number of elements in an
Array is myArray.length.
```

15. Deleting elements

To remove an element from an Array at index i, use the following code, where deleteCount is the number of elements to remove:

```
myArray.splice(i, deleteCount);
```

You can use the same method to add elements by setting the value of deleteCount to 0 and then passing further parameters. In this instance, the element "Flash" will be added at index 1.

```
myArray.splice(1, 0, "Flash");
```

16. Multidimensional Arrays

```
var subArray1 = new
Array("Macromedia", "Flash",
"MX", "2004");
var subArray2 = new
Array("Macromedia", "Director",
"MX", "2004");
var myArray = new Array("Hello",
subArray1, subArray2);
```

Using the above myArray[0][1] is 'Flash': 0 to choose the first element of myArray and then 1 to choose the second element of the embedded Array.

17. Joining an Array

You can join an Array together and return it as a String, a line of text. Each element is separated from the previous one with the parameter passed to the method join.

```
var myArray = new
Array("Macromedia", "Flash",
"MX", "2004");
trace(myArray.join(" "));
```

In this instance, 'Macromedia Flash MX 2004' is sent to the Output Window.

18. Creating an Array from a String

```
str = "Macromedia Flash MX
2004";
myArray = str.split(" ");
```

Using the split method of a String object, you can create an Array. You pass the symbol used to break the String into separate elements – in this case, a space character. The result of this is an Array of four elements: "Macromedia", "Flash", "MX", "2004".

19. Variable scope

Flash can contain MovieClips embedded into MovieClips. The root level timeline is itself a MovieClip. If you have a variable at the root called x then this is different from a variable called x inside a MovieClip called mc.

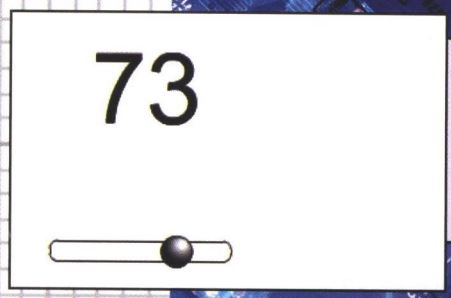
To access variables in the MovieClip mc from the root use: mc.x

To access variables in the root from mc use: _root.x or this._parent.x

20. PC or Mac

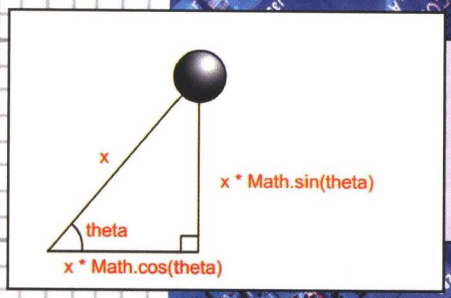
To find out if the *Flash* movie is running on a PC or a Mac use:

```
os = $version;
if (os.substr(0, 1)=="W"){
  os = "Windows";
}else{
  os = "MAC";
}
```



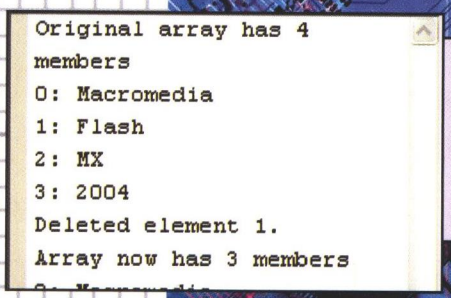
TIP 09

You can link a text box to a scroll bar in the Properties window.



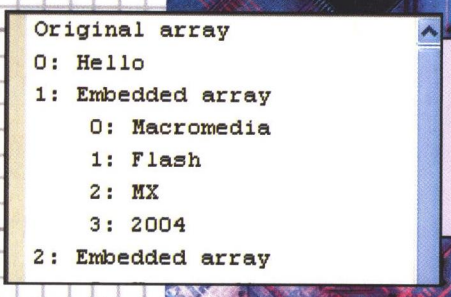
TIP 10

The Math object methods sin and cos enable movement in a circle.



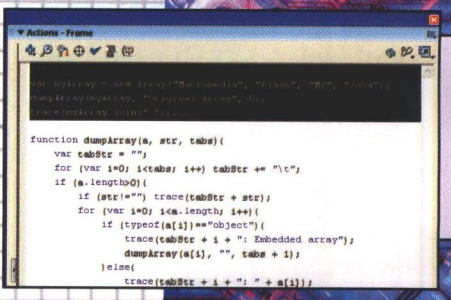
TIP 15

Use code to delete elements from, and add elements to, an Array.



TIP 16

Multiple elements can be added to create multidimensional Arrays.



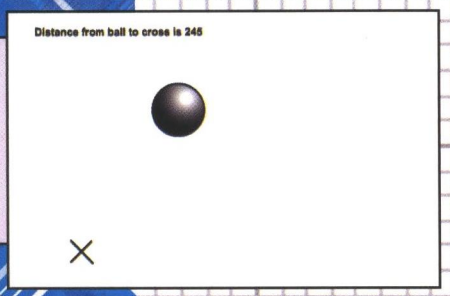
TIP 17

Arrays can be joined together to create a String – that is, a line of text.

60 ACTIONSCRIPT TIPS

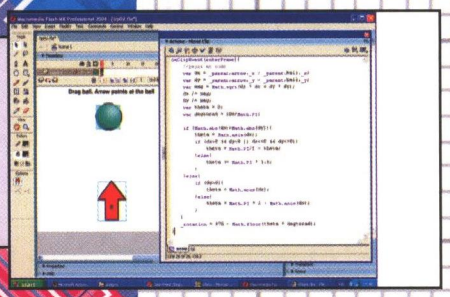
TIP 1

Work out the distance between two clips with some basic calculations.



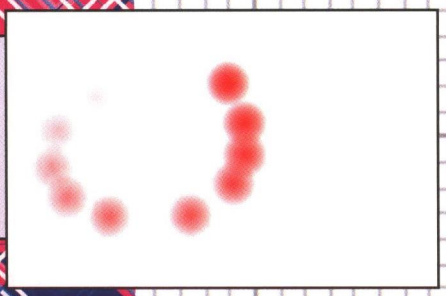
TIP 2

Determine the angle between the two clips before rotating the arrow.



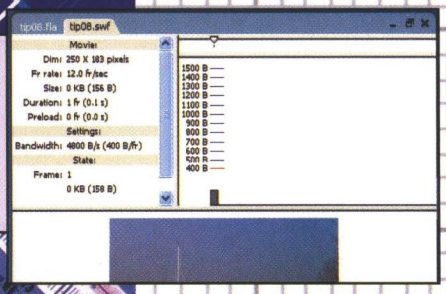
TIP 3

Make trails by using two MovieClips: individual and overall controller.



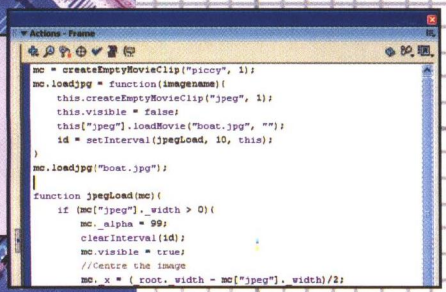
TIP 8

Create an empty MovieClip to make loading JPEG images easy.



TIP 9

A progress indicator enables you to see how long it takes JPEGs to load.



1. Calculating distances

You can calculate the distance between two MovieClips.

```
var dx = _parent.ball._x - _
parent.cross._x;
var dy = _parent.ball._y - _
parent.cross._y;
var dist = Math.sqrt(dx * dx +
dy * dy);
```

Use the square root of the sum of the square of the distance apart in the x and the square of the distance apart in the y.

2. Point one clip at another

This tip first creates two variables, dx and dy, which are the distances apart of the two clips in the x and y axes. Then you get the distance between these points. Next, divide dx and dy by this distance to get what is called a unit length vector. You can use this vector to determine the angle between the two clips and hence rotate the arrow clip to point at the ball.

3. Creating trails

Use two MovieClips: one clip is the individual part of the trail and the second is the overall controller. In this example, the individual part is set to fade out over time using the _alpha property. The controller creates new clips, starting at level 1, on the fly, at the existing mouse position. Once the limit of clips is reached, the controller loops back to level 1.

4. Using loadVariables

```
loadVariables("tip04.txt", this);
```

The above code segment loads the contents of the file 'tip04.txt' from the cover CD into named variables at the _root of the Movie. tip04.txt is simply:

```
&count=100&
&loaded=true
```

Each variable name/value pair is separated by an ampersand.

5. Using the LoadVars object

```
lc = new LoadVars();
lc.onLoad = function(success){
  trace(success + " " + this.
count);
}
lc.load("tip05.txt");
```

The LoadVars object is a more elegant way of doing the same thing as in the previous tip. The principal benefit is you can define an onLoad function that's called when the data is loaded. To start the load, call the load method with a single parameter, the file to load.

6. Using the name Array

Every MovieClip has an array of names that can be used to access variables and Objects.

```
//Access any variable or Object
within the MovieClip
this[myName];
//Access any variable or Object
within the MovieClip s parent
this._parent[myName];
//Access and variable or Object
within the _root timeline.
_root[myName];
```

7. Using setInterval

To define something that occurs at a certain time interval or a timeout event, use setInterval. You pass a function that is called when the number of milliseconds in the second parameter elapses.

```
var id = setInterval(timeOut,
1000);
```

Use the return value from setInterval to clear the event using:

```
clearInterval(id);
```

8. Loading a JPEG image

Just create an empty MovieClip and load the image using the code below:

```
mc = createEmptyMovieClip("piccy
", 1);
mc.loadjpg =
function(imagename){
  loadMovie("boat.jpg", "");
}
mc.loadjpg("boat.jpg");
```

9. Adding a progress indicator

Use the following code to add a progress indicator to the JPEG Loader.

```
mc = createEmptyMovieClip("piccy
", 1);
mc.loadjpg =
function(imagename){
  this.createEmptyMovieClip("jpe
g", 1);
  this.visible = false;
  this["jpeg"].loadMovie("boat.
jpg", "");
  id = setInterval(jpegLoad, 10,
this);
}
mc.loadjpg("boat.jpg");
```

The trick here is to create an empty MovieClip, called 'jpeg' within the MovieClip mc. Set the MovieClip mc to invisible and then load the jpeg into the sub clip. Set an interval caller so that you can use this to update the progress. Take a look at the code on the CD for details.

10. Opening a file requester

This code opens a file requester:

```
os = $version.substr(0, 1);
if (os == "W"){
 getURL(".");
}else{
 fscommand("exec", "browseCD");
}
```

11. Repeating sections of code

The onEnterFrame function is useful for sections of code that need to repeat at the pace of the movie's fps. You can create them dynamically using:

```
this.onEnterFrame = function(){
//Things to do each frame
}
```

12. Starting with the drawing API

The code below creates a MovieClip and then uses the drawing API to draw a black line that's two pixels thick from the point (275, 200) to the current mouse position.

```
this.onEnterFrame = function(){
 removeMovieClip("mc");
 mc = this.createEmptyMovieClip
("mc", 0);
 mc.lineStyle(2, 0x000000,
100);
 mc.moveTo(275, 200);
 mc.lineTo(_root._xmouse, _
root._ymouse);
}
```

13. Draw rectangle function

Flash has the ability to extend the functionality of the built-in objects. You can add a prototype, in this case a function to draw a rectangle, to the MovieClip object and then every MovieClip has the additional function, drawRect.

```
MovieClip.prototype.drawRect =
function(l, t, w, h, scol, fcol){
 this.lineStyle(1, scol, 100);
 this.beginFill(fcol, 100);
 this.moveTo(l, t);
 this.lineTo(l + w, t);
 this.lineTo(l + w, t + h);
 this.lineTo(l, t + h);
 this.lineTo(l, t);
 this.endFill();
}
```

14. Drawing a curve

Set the starting position of a curve using the drawing API method moveTo. Then determine two points: the end point and a control point that determines the bend.

```
 mc = createEmptyMovieClip
("curves", 1);
 mc.lineStyle(2, 0xFF0000,
100);
 mc.moveTo(10,200);
 mc.curveTo(275, 0, 540, 200);
```

15. Drawing a circle

Use the drawing API method, curveTo, seen in the previous tip. Build a circle in 45-degree sections. Setting the end points uses the same method as beginners tip 10. The control points are halfway between the start and end points.

16. Drawing a pie chart

Extend the drawCircle method to create drawCircleSegment. Then generate a batch of data and use this to draw a pie chart. The code for this tip is on the cover CD.

17. Embedding fonts

Create a dynamic text box at design time and set this up to include the font outlines needed for the dynamically created text fields. Once the movie has the outlines required, set up a text field using createTextField to use the outlines just by setting the value for embed fonts to true.

```
newTextField.embedFonts = true;
```

18. Dynamic masking

Here, we create an empty clip and draw a rectangle in it. Then we create a second empty clip setting the first clip as a mask. By setting the mask as the drag item and loading an image into the second clip, we can move a window over the image.

```
mask = createEmptyMovieClip("box
", 2);
mask.drawRect(0, 0, 100, 100,
0xFF0000, 0x00FF00);
mc = createEmptyMovieClip("piccy
", 1);
mc.setMask(mask);
```

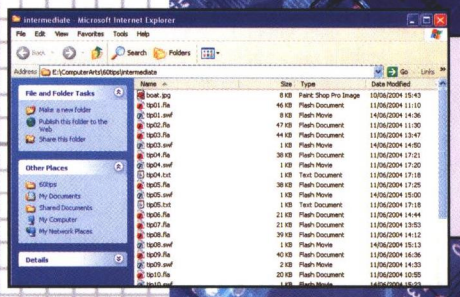
19. Using #include

You can write ActionScript in a text editor or use *Flash MX 2004 Pro* to add an ActionScript file (.as). To use the ActionScript in your movies, use this:

```
#include "tip19.as"
```

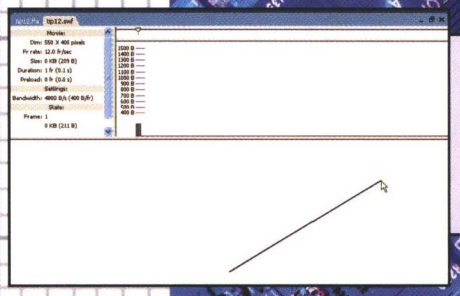
20. Email validation

The ActionScript File 'tip20.as' on the cover CD contains the code to validate an email address. Using simple String parsing, it checks if the '@' and '.' symbols are present by ensuring that there's a name before the '@' symbol and a domain after it. The function getValidationCode returns a value showing any errors that may have occurred.



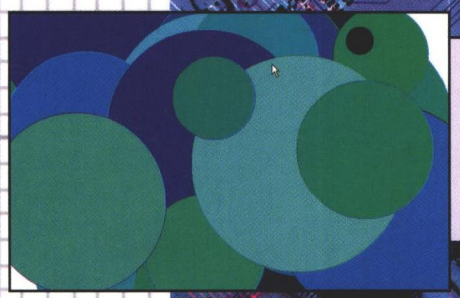
TIP 10

Enable users to browse files using this handy piece of code...



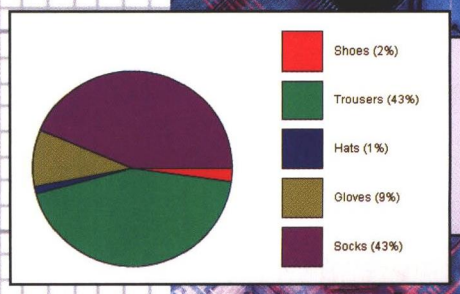
TIP 12

Dynamically draw shapes using the *Flash* ActionScript drawing API.



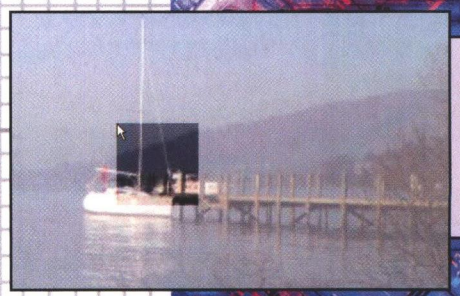
TIP 14

Draw a curve using the drawing API method moveTo.



TIP 16

Create a pie chart including a colour guide and text fields.



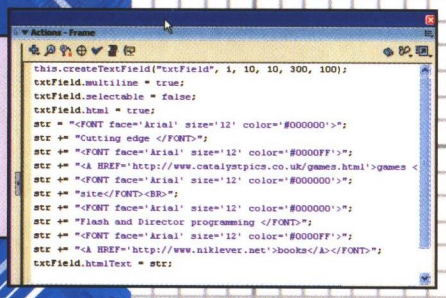
TIP 18

Dynamically mask an image by drawing a shape in an empty clip.

60 ACTIONSCRIPT TIPS

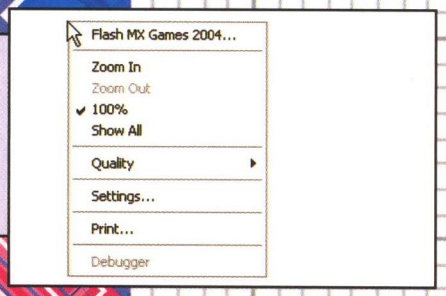
TIP 2

You can add HTML formatted text by setting the text field to an HTML field.



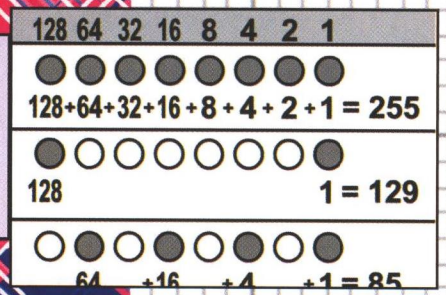
TIP 3

Creating a ContextMenu object enables you to add items to the menu.



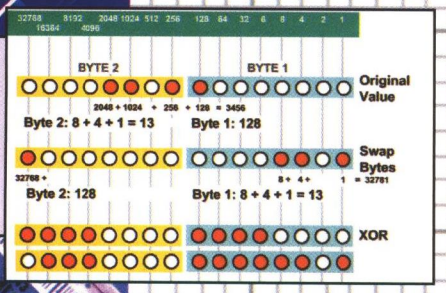
TIP 5

Manipulate data at bit level using bitwise operators.



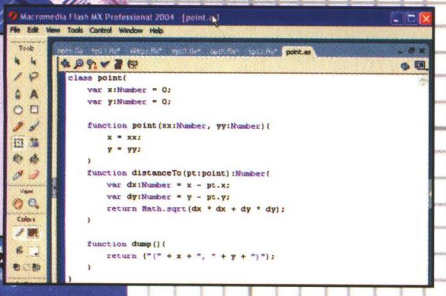
TIP 7

Bitwise operators can be used to encode info you pass over the Web.



TIP 12

Get wise to using classes within an ActionScript file.



1. Additional TextField formatting

There are many properties of a text field created with createTextField. You can add a border using:

```
txtField.border = true;
```

You can size the border to fit the current text using:

```
txtField.autoSize = true;
```

You can set its type to an Input Field using:

```
txtField.type = "input";
```

2. Using HTML text

By setting the text field to be an HTML field, you can add HTML formatted text. In this example, we add two lines each with formatted text and a Hyperlink. The txtField has selectable set to false to make sure the correct cursor is displayed, and it has multiline set to true.

3. Altering the Context menu

To add items to the Context menu, you first need to create a custom ContextMenu object. Then you use the customItems Array to add content to the menu:

```
myArray.push(myContextMenuItem);
```

A ContextMenuItem is created using two parameters — the text to display and function to call. Finally, you set the context menu as the menu property of an Object on screen, in this case the _root Object.

4. Using multiple Context menus

You can attach a ContextMenu object to a specific button, MovieClip, or text field object, or to the entire movie. Use the menu property of the Button, MovieClip, TextField or _root classes to do this.

5. Using bitwise operators

At the most basic level, the data stored on a computer breaks down to either 1 or 0. This is a bit. You can manipulate data at the bit level using the bitwise operators.

```
n <<= 1;
```

The above code has the effect of moving the bits in n one to the left, which is the same as doubling its value.

6. Bitwise and, or and xor

For these operations, the bits in two variables are compared individually:

And (&) — if both bits are on, the resulting bit is set to on.

Or (|) — if either bit is on, the resulting bit is set to on.

Xor (^) — if either bit is on but not both, the resulting bit is set to on.

7. Encoding and decoding

Using the bitwise operators it's possible to encode information that you pass across the Web. For example, if a variable can only have values up to 65535 then it can be stored in two bytes. Get the first byte and the second byte and then swap them round. Xor the result and you have a garbled value. To restore it, do the operation backwards.

8. Using Listeners

Assuming that txtField is an Input Text Field then changing the content will have the result of calling the onChanged method of the Listener Object.

```
myListener = new Object();
myListener.onChanged =
function(){
  trace("Contents changed");
}
txtField.
addListener(myListener);
```

9. Responding to code events

When the text of a TextField is set in code, the onChanged event is not triggered. To get around this, you can add a setText method to the TextField Object.

```
TextField.prototype.setText =
function(str){
  this.text = str;
  this.broadcastMessage("onCha
nged");
};
```

10. Creating a point class

Classes are an excellent way of keeping your code well structured. They are easily created by writing a constructor function:

```
function point(){
  this.x = 0;
  this.y = 0;
}
```

To create a point simply call:

```
pt = new point();
```

The object will be created with the variables x and y set to zero.

11. Adding methods

You can add methods to the point class by using the prototype keyword followed by a method name.

```
point.prototype.init =
function (x, y){
  this.x = x;
  this.y = y;
};
```

12. Using a class file

An alternative approach is to create an ActionScript file and then create the class in the file. You declare any member variables and write a constructor function that has the same name as the class.

```
class point{
  var x:Number = 0;
  var y:Number = 0;

  function point(xx:Number,
yy:Number){
        x = xx;
        y = yy;
  }
}
```

13. Externally declared classes

To use an externally declared class, the class file needs to be in the same folder as the project file.

```
v1 = new vector(0, 1);
v2 = new vector(1, 1);
trace("Angle between " +
v1.dump() + " and " +
      v2.dump() + " is " +
v1.angleBetween(v2) + "
degrees");
```

14. Making class variables private

The keyword 'private' means that the direct value of a variable in the class is not exposed. You can write methods to get and set them. It makes the code more robust.

```
class triangle{
  private var pt1:point;
  private var pt2:point;
  private var pt3:point;
  private var triarea:Number =
0;
  ...
```

15. Using simple physics

The bat moves and rotates using basic physics calculations. First, create a point 100 pixels up from the bat's centre point. Then calculate a force based on this position and the current mouse position. This is used to move the bat. Finally, we calculate a rotational force based on the current movement force.

16. Starting with XML parsing

The XML object is useful for accessing xml feeds from the Web. While powerful, it can be a little confusing. The purpose of this small example is to generate a list of data from any XML feed so that you can see the names and values of the many nodes. This takes a feed from a free news supplier.

17. Generating HTML from an XML feed

Once you've examined the feed, you can extend the parsing to generate anything your movie requires. In this example, we convert the news feed into HTML with hyperlinks. Only the nodes that have the nodeName, item, have the actual data required.

18. Using the LocalConnection object

The LocalConnection object is useful for communicating between two swfs running on the same PC. In this example, we create a receiving movie (tip18a.fla) and a sending movie (tip18b.fla). The sender communicates with the receiver using the connect name; it can then call methods on the receiver.

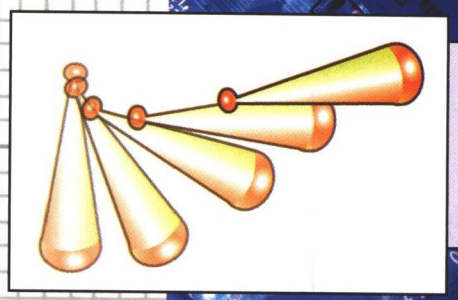
19. Using the SharedObject

The SharedObject is a good alternative to using cookies. You can store data on the client's machine or the remote server that can be accessed later.

```
so = SharedObject.
getLocal("myInfo");
if (so.data.favoriteColor==und
efined){
  //No saved data
  so.data.favoriteColor =
"orange";
  colors = new Array("red",
"blue", "green", "blue");
  so.data.possibleColors =
colors;
}else{
  //Show the saved data
  trace(so.data.
favoriteColor);
  trace(so.data.
possibleColors);
}
```

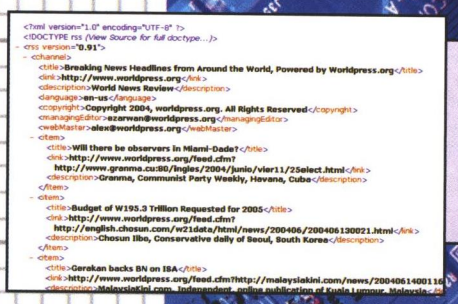
20. Using the ASV application

If you were to lose your project but still had the swf, you may well be able to recover the code using the *ActionScript Viewer*, which is a terrific application. The most current demo version of ASV can always be found at http://www.swf2fla.com/asvdemo.zip. ca-p



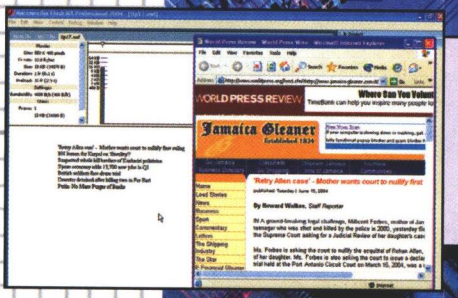
TIP 15

Move and rotate an object using simple physics calculations.



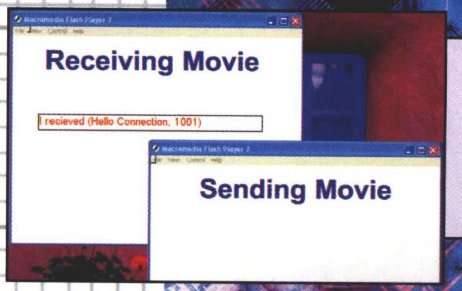
TIP 16

Access XML feeds from the Web using the XML object...



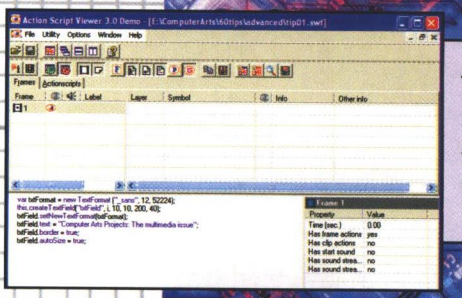
TIP 17

...and then use the feed to generate anything your movie requires.



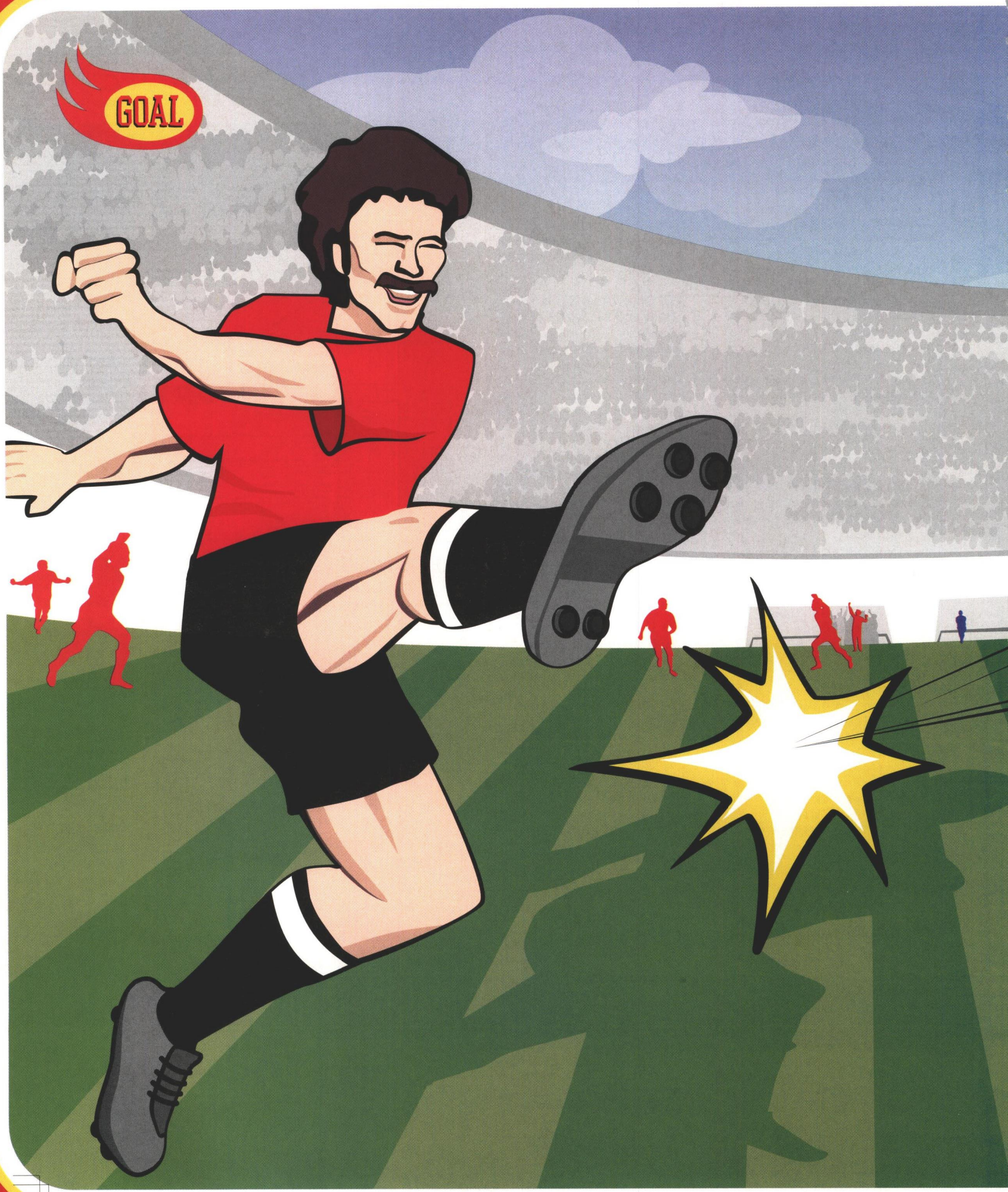
TIP 18

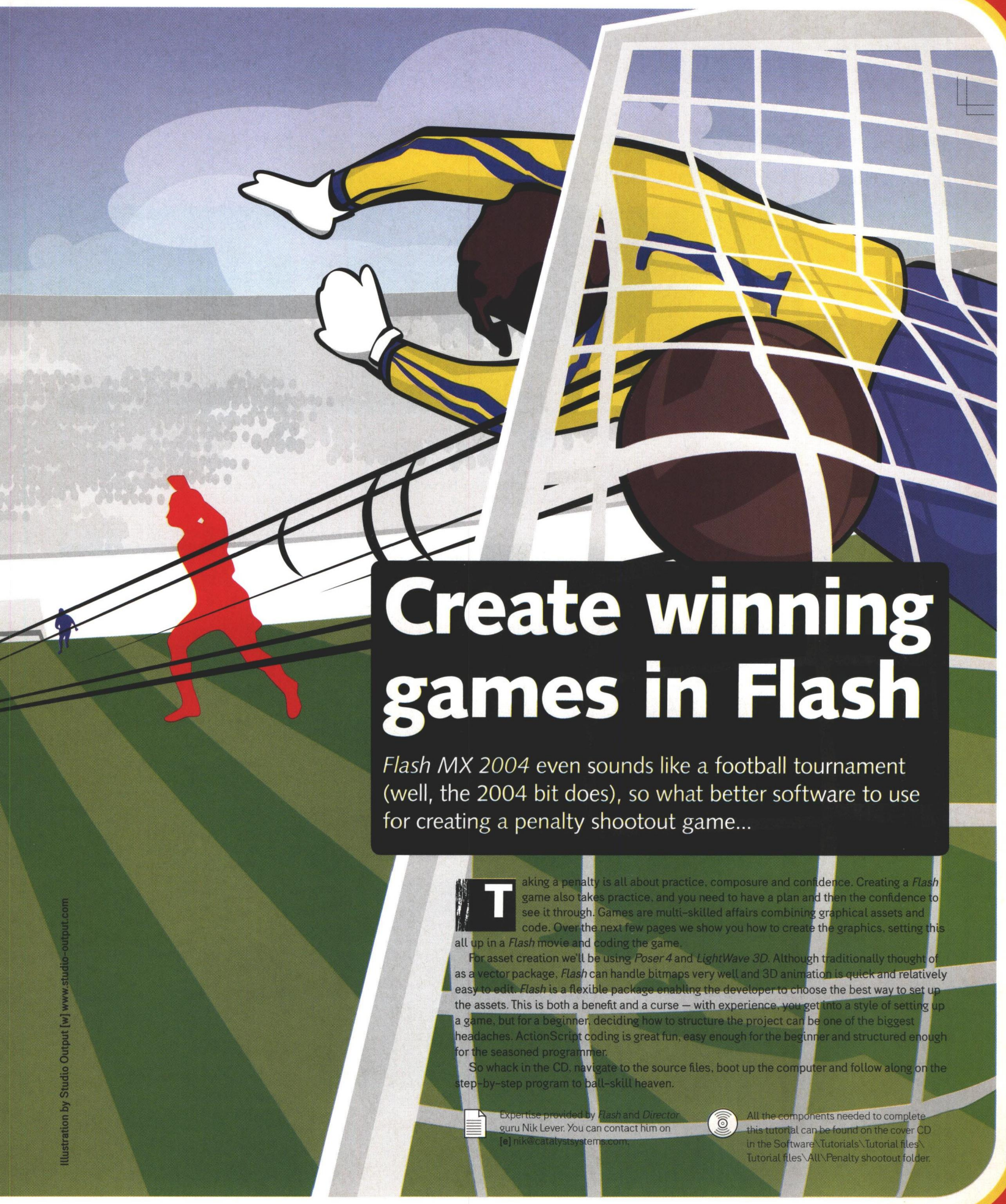
Communicate between two SWF movies on the same machine.



TIP 20

The *ActionScript Viewer* can recover code from lost projects.


GOAL



Create winning games in Flash

Flash MX 2004 even sounds like a football tournament (well, the 2004 bit does), so what better software to use for creating a penalty shootout game...

Illustration by Studio Output [w] www.studio-output.com

Taking a penalty is all about practice, composure and confidence. Creating a *Flash* game also takes practice, and you need to have a plan and then the confidence to see it through. Games are multi-skilled affairs combining graphical assets and code. Over the next few pages we show you how to create the graphics, setting this all up in a *Flash* movie and coding the game.

For asset creation we'll be using *Poser 4* and *LightWave 3D*. Although traditionally thought of as a vector package, *Flash* can handle bitmaps very well and 3D animation is quick and relatively easy to edit. *Flash* is a flexible package enabling the developer to choose the best way to set up the assets. This is both a benefit and a curse – with experience, you get into a style of setting up a game, but for a beginner, deciding how to structure the project can be one of the biggest headaches. ActionScript coding is great fun, easy enough for the beginner and structured enough for the seasoned programmer.

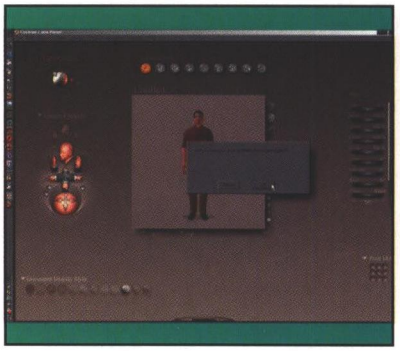
So whack in the CD, navigate to the source files, boot up the computer and follow along on the step-by-step program to ball-skill heaven.

Expertise provided by *Flash* and *Director* guru Nik Lever. You can contact him on [e] nik@catalystsystems.com.

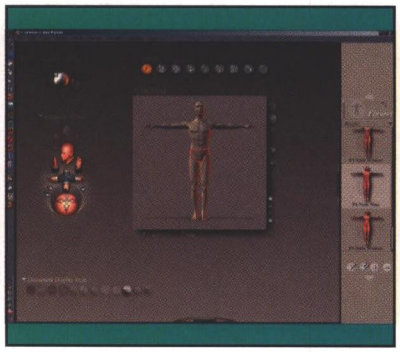
All the components needed to complete this tutorial can be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Penalty shootout folder.

Part 1 : Creating the assets

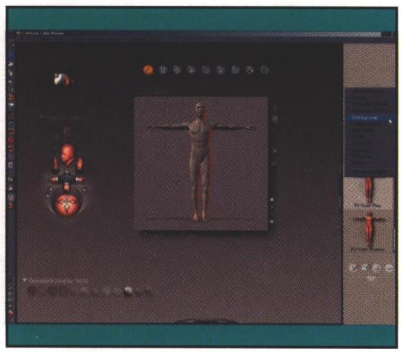
Sprite graphics can be created easily using *Poser* and *LightWave 3D*...



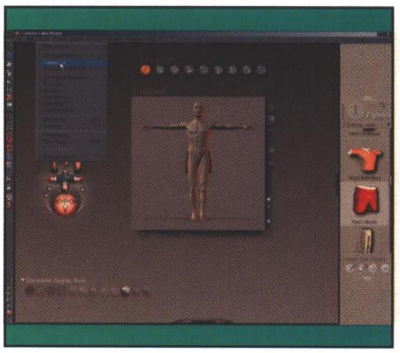
1 To create the player and goalkeeper we're going to use a combination of *Poser 4* and *LightWave 3D 7.5*. First, run *Poser 4* and select File>New. This creates a default *Poser* scene featuring the default male character. Select the hips of the man by clicking on them, and press delete.



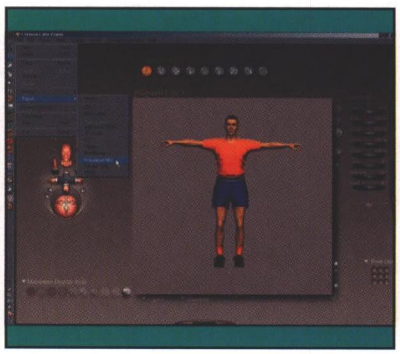
2 You should now have an empty project. Click the side bar device at the right and select P4 Nude Man from the icons. Using the buttons just below the icons, click the one with two ticks – Create New Figure. This adds the nude figure to the project.



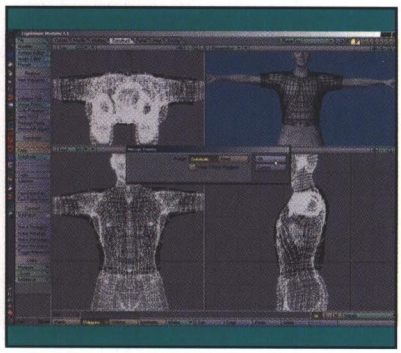
3 Again using the side bar, click on the word People and from the menu select Clothing-male. You need to add shorts and a shirt to the character, so select Men's shorts and again click Create New Figure. Repeat this process to add your choice of shirt.



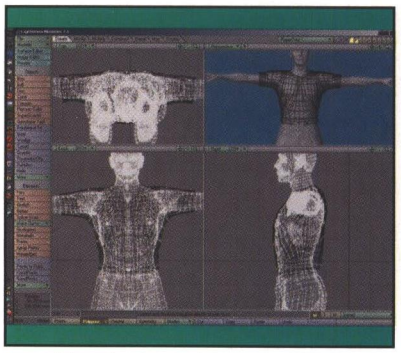
4 At this stage, the clothes will not conform to the character – you must tell *Poser* that this is what you require by using the menu option Figure>Conform To... This brings up a dialog box where you select which object you want the clothing to fit. Select Figure 1 and click OK.



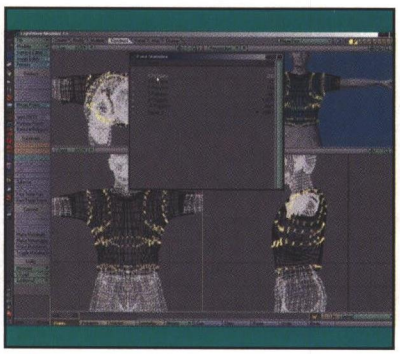
5 You might find it quicker and easier to open the file player.pz3 from the Penalty shootout\3D\Poser folder on the cover CD. You should now have a completed figure to export to *LightWave*. Choose File>Export>Wavefront OBJ. In the next dialog, select Single frame and save using the default settings.



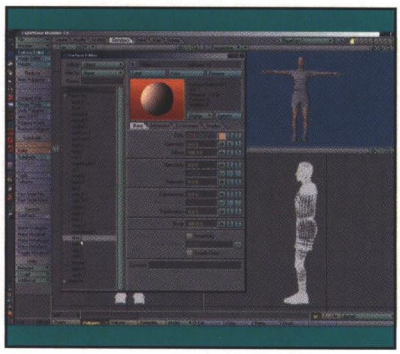
6 You'll need to do some 'tidying-up' before you can use the model for animation. Start by removing unwanted points. Select Construct>Merge Points. Select Automatic and unselect 'Keep 1-point polygons' then press OK.



7 Click on the shirt and press x to cut it out, then move to an alternative layer and press v. Select layer 1 as the foreground and layer 2 as the background, highlight the torso polygons beneath the shirt and press delete to remove them.

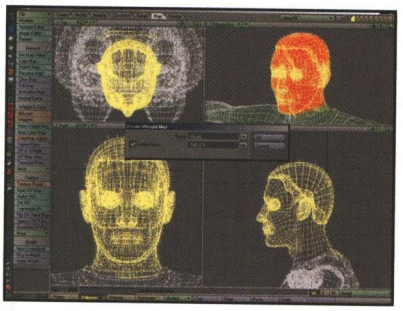


8 Having deleted the torso polygons, you might find you have stray points. Select Point selection mode at the bottom of the screen and press w to bring up the Point Statistics window. Click + to the left of 0 Polygons, and press delete.



9 Repeat Steps 7 and 8 for the polygons under the shorts, boots and socks. Then use the Surface Editor to set the colours and texture detail. The Export/Import process retains surface names but loses the surface detail.

10 To speed things up, load in the object 'player.lwo' from the Penalty shootout\3D\Objects folder on the cover CD. Here is the completed player. In layer 2, create the bones to animate the character using Create>Skelegons. Give each bone a sensible name using Detail>Rename Skelegon.

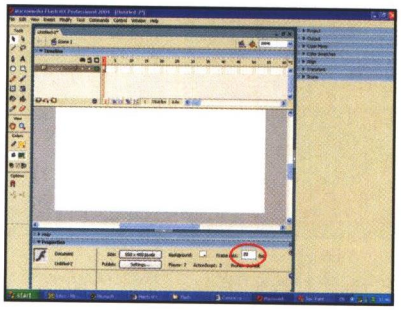


11 The points a bone affects can be assigned using weight maps. Select a number of vertices and then use Map>New Weight Map. Name the map. Choose Weight Map Shade from the Perspective window dropdown menu to view the weight maps. Save the object then move to the animation and rendering application: Layout.

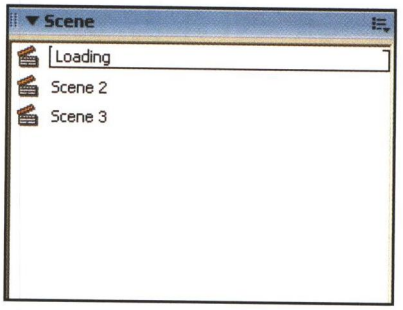
12 Select Items>Add>Bones>Convert Skelegons to Bones. Select the bone button at the bottom to move parts of the player's body. Use the keyframe tools to animate, then press L to create a limited render region. Drag the dotted yellow box until it forms the minimum area for the player's movement. Render the scene saving as 32-bit pngs.

Part 2 : Organising the assets

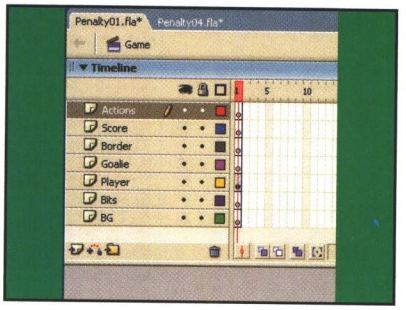
If you're new to *Flash*, deciding how to structure your game will be one of the most difficult decisions...



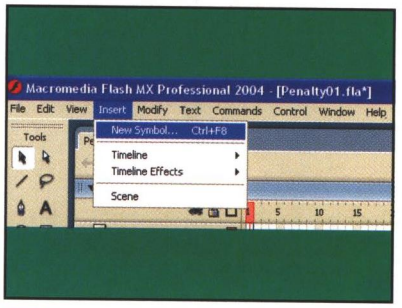
1 Create a new document by choosing File>New. Then select Flash Document from the General tab of the next dialog box. Open the Movie Properties window by selecting Window>Properties and then clicking in an empty area of your Movie. Set the framerate to 20 fps.



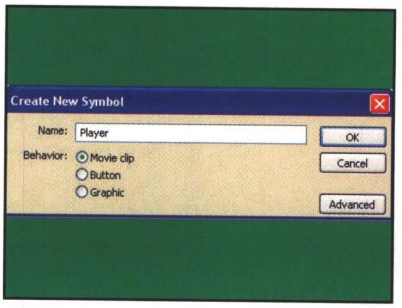
2 A default document has a single scene but we need three. So choose Insert>Scene twice to add two extra scenes. Then open the Scene Window by selecting Window>Design Panels>Scene. Using the Scene Window you can rename the Scenes to Loading, Game and Game Over.



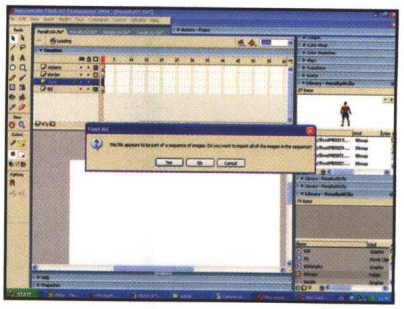
3 Loading and Game Over scenes will have the layers BG, Text, Border and Actions. The Game scene uses the layers BG, Bits, Player, Goalie, Border, Score and Actions. Add layers by clicking the bottom-left icon in the Timeline Window. Rename the layer by double-clicking the name.



4 Player, Goalie, PowerBar, Ball and BG are all MovieClips. A *Flash* MovieClip is a separate movie that contains its own timeline; multiple timelines give *Flash* its power as a development tool. To create a MovieClip, select Insert>New Symbol... and choose Movie Clip.



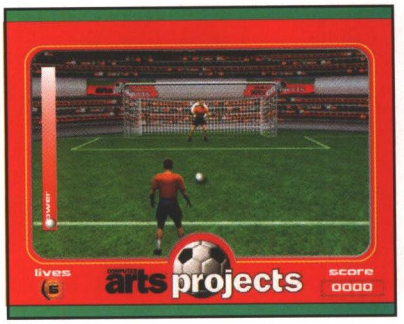
5 Create a new MovieClip and name it Player. Newly created symbols will appear in the Library. Choose Window>Library to open the Library window. Then drag the Player clip onto the stage. Double-click the dot that indicates an empty MovieClip.



6 We want to add the images we created at Step 12 (above) to this clip. Select Import>To Stage... and locate the first image in the animation. *Flash* will spot this as a sequence. In the following dialog, choose Yes to 'Do you want to import all the images in the sequence?'

Part 3: Adding the code

It's time to start turning the graphics into a game using ActionScript...

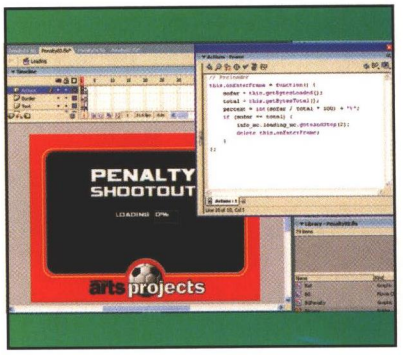


1 The background, ball and goalkeeper are also *LightWave* models. Import these in the same way, making each one a separate and named MovieClip before you begin to work on the code. The final step in creating the assets is to draw the borders and score graphics.

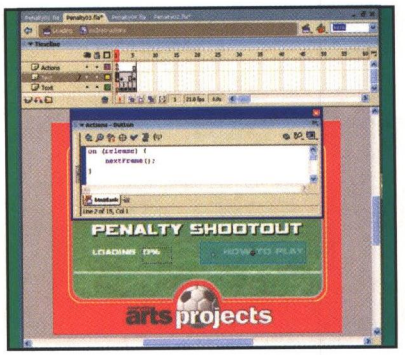
2 Penalty02.fla on the cover CD is a *Flash* document containing all the assets but no code. The first step is to create animation loops inside the various MovieClips. The Player clip has two actions: Shuffle, for moving from side to side; and Shoot, to kick the ball. Add an Actions layer.

3 Add a Shuffle frame label to frame 2, and a Shoot label to frame 11. Click on frame 2, and in the Properties window type Shuffle in the Frame Edit box. Shuffle is a looping animation, so go to frame 10 and press F7 to insert a blank keyframe. Press F9 to open the Actions window and type:

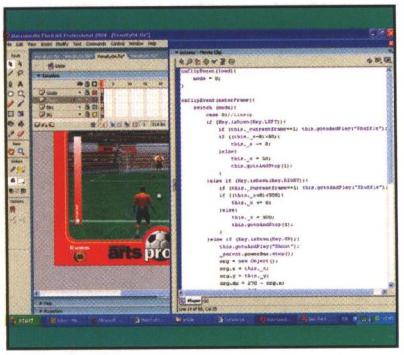
```
gotoAndPlay("Shuffle");
```



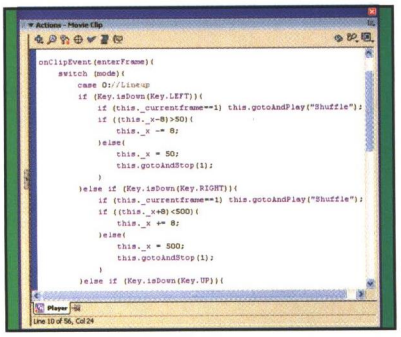
4 The indicator to show how much has been downloaded in this game is handled by using an onEnterFrame script defined on the Actions layer frame 1. This derives a percentage loaded by dividing the bytes downloaded by the total number of bytes in the swf.



5 The Loading scene also contains the instruction. The script for the 'How to Play' button can be seen by opening Penalty03.fla from the cover CD, going to the Loading scene, double-clicking the turquoise area on frame 2 and looking at the Actions window.

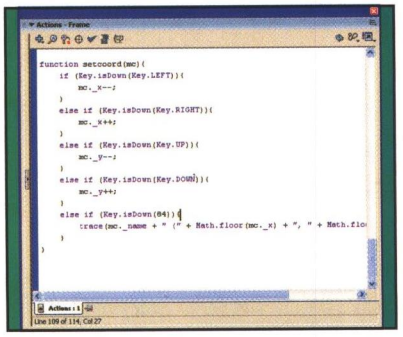


6 Player motion is handled using an onClipEvent(enterFrame) handler. This event uses the Key object to detect key presses. To view the code, open Penalty04.fla from the cover CD, select the 'Game' scene and click on the player. The script will be seen in the Actions window.



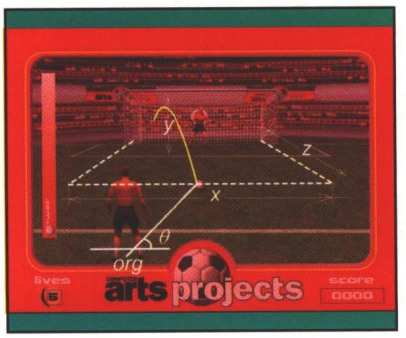
```
onClipEvent(enterFrame){
    switch (mode){
        case 0://Line up
        if (Key.isDown(Key.LEFT)){
            if (this._currentframe==1) this.gotoAndPlay("Shuffle");
            if ((this._x-8)>50){
                this._x -= 8;
            }else{
                this._x = 50;
                this.gotoAndStop(1);
            }
        }else if (Key.isDown(Key.RIGHT)){
            if (this._currentframe==1) this.gotoAndPlay("Shuffle");
            if ((this._x+8)<500){
                this._x += 8;
            }else{
                this._x = 500;
                this.gotoAndStop(1);
            }
        }else if (Key.isDown(Key.UP)){
```

7 There are three possible mode settings for the player: 0 for line up, 1 for the run up, and 2 after the kick. The script checks the Left, Right and Up arrows if the current mode is 0. Left and Right are used to move the player if the player will stay within bounds.

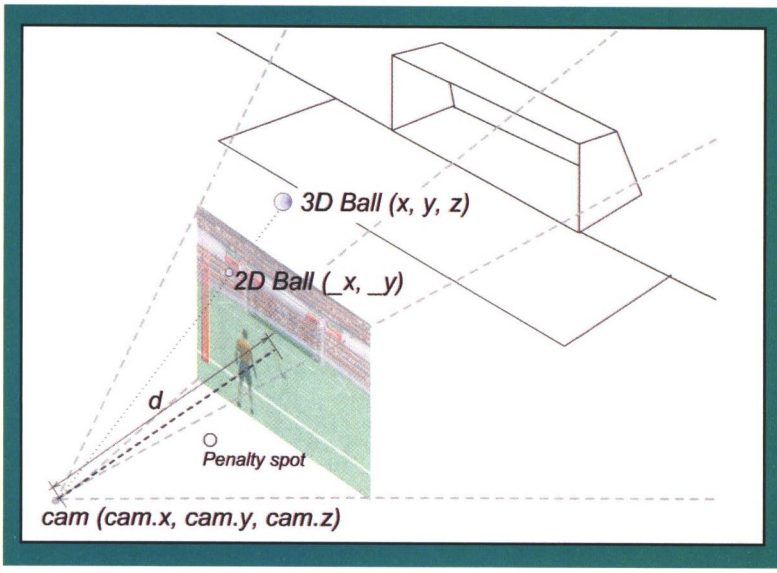


```
function setcoord(mc){
    if (Key.isDown(Key.LEFT)){
        mc._x--;
    }
    else if (Key.isDown(Key.RIGHT)){
        mc._x++;
    }
    else if (Key.isDown(Key.UP)){
        mc._y--;
    }
    else if (Key.isDown(Key.DOWN)){
        mc._y++;
    }
    else if (Key.isDown(84)){
        trace(mc._name + " (" + Math.floor(mc._x) + ", " + Math.flo
    }
}
```

8 The setcoord function, defined in frame 1 of the Actions layer for the Game scene, is a function for moving a MovieClip. The function takes a MovieClip, mc, as a parameter and then uses the arrow keys to move it and the T key to send its current position to the Output window – useful for determining the bounds of a moving clip.



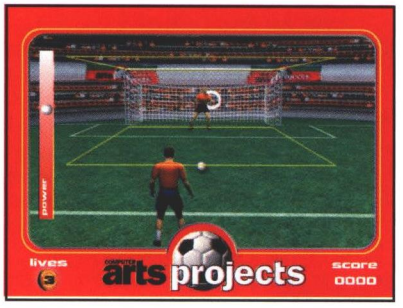
9 Pressing the Up arrow triggers the kick. Because of the viewing angle the kick is handled as a 3D problem. The kick angle is θ, and the strength of the kick is defined using the '_currentframe' of the power bar MovieClip. The problem reduces to just these two parameters, which are passed to a function 'kick' defined in 'ball'.



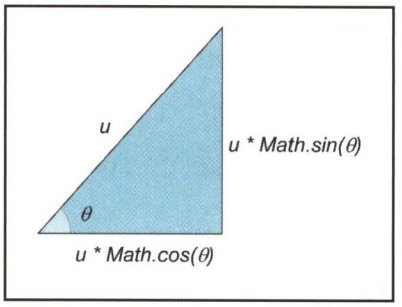
Adding a third dimension

To use 3D we add a third coordinate, z, to the x and y positions. We can define the location of the goal using values of x to define left and right and values of y to define up and down. In addition, the position back from the ball kick location is defined using z. But we also need to be able to convert these 3D coordinates back into screen coordinates so we can position the ball. This is surprisingly easy. First, we need to know the camera's location (cam.x, cam.y, cam.z) and the distance of the camera to the screen, d. Then the location is defined as:

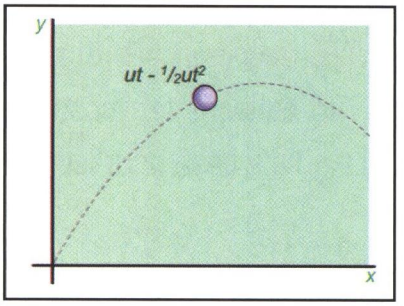
```
_x = ((x - cam.x) * d)/(z - cam.z) + cam.x;
_y = ((y - cam.y) * d)/( z - cam.z) + cam.y;
```



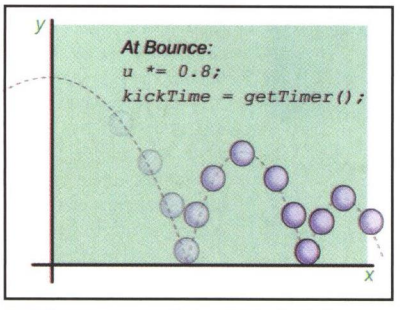
10 When using 3D-to-2D conversions in this way, the position of the camera is crucial to the success. The function drawRect defined in the first frame of the Actions layer is designed to draw a rectangle using the 3D conversion. This needs adjusting until it matches the ground perspective.



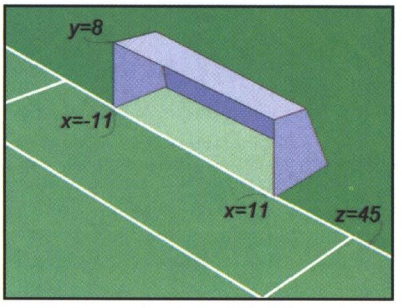
11 The ball is moved in the x and z in 3D space for each frame, using the variables dx and dy calculated in the kick function. They are calculated from the angle of the kick based on simple trigonometry. The side near the angle uses Math.cos and the side opposite Math.sin.



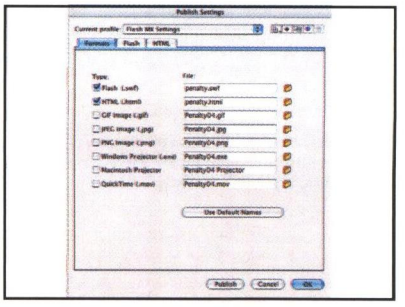
12 Projectile motion takes a launch power 'u'. The height of the ball at time 't' is defined as ut – 1/2gt2 where g is gravity. To convert this into useful values for your game takes practice. Adjust the time taken to start falling as indicated in code and adjust the speed of the fall using the value for gravity.



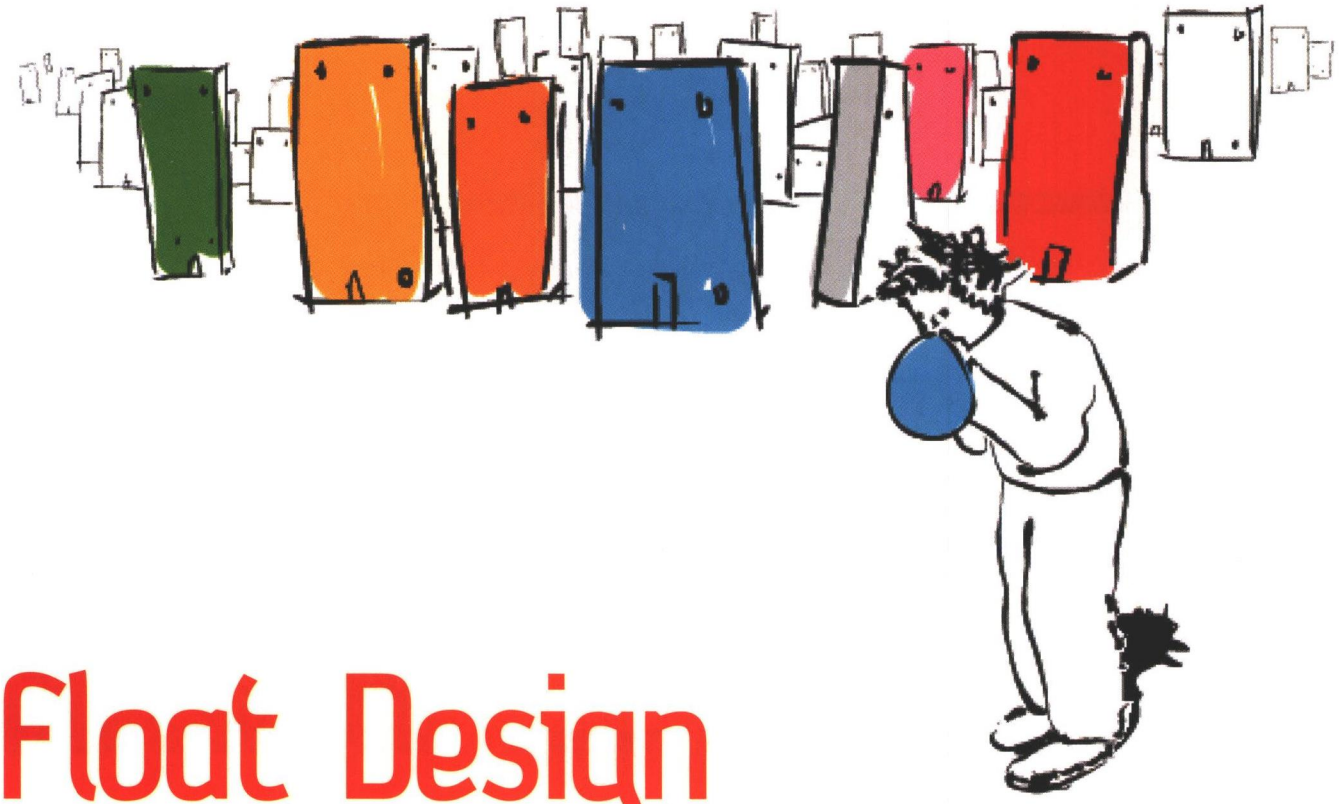
13 The advantage of using 3D for the ball location is that the problem is made easier. A value of y of zero means we need to bounce the ball. Which involves reinitialising 'kickTime' and then simply reducing u, dx, and dy by 20 per cent, (by multiplying them by 0.8).



14 The ball reaches the goal when the z value is 45. A goal is successful if -11<x<11 and 0<y<8. This is tested in the onClipEvent(enterFrame) handler for the ball. Then a new penalty is initiated using the newPenalty function defined in frame 1 of the Actions layer.

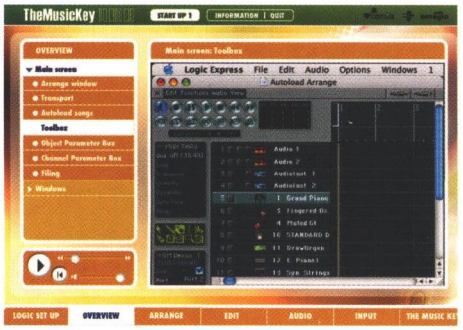


15 Save your document, publish the SWF or both SWF and HTML by going to File>Publish Settings and there you have it – a 3D game that uses a bit of mathematic theory in 36 steps! ca-p

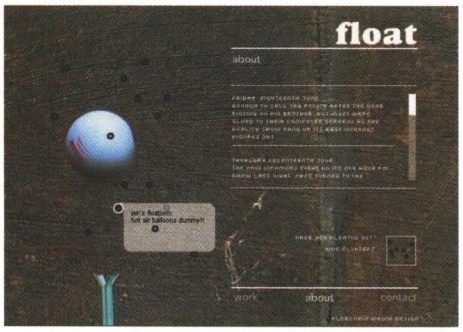


Float Design

"What ever happened to Tim Ketterer?" was a message attached to 100 balloons let loose for a fine art project seven years ago. Lately he's been spotted founding a Web design company in Bath...



TheMusicKey is one of Float's main ongoing projects. Put together in *Flash* and *Director*, it's a series of training discs helping music teachers with emagic's *Logic* music software.



Float Design's own Website, [w] www.floatdesign.net, is currently being redesigned, and it will pull in some aspect of leaving and picking up anonymous messages.

When Tim Ketterer's Balloon Sender site won the Mando Student Website Design Award in 2003 he was fending off competition from all around the country, but also from a colleague on his course, Stephen Lockhart. Both were at Bath Spa Univerity studying for an MA in Interactive Media. Both were finalists in the awards. And both were collaborating together on projects for the course.

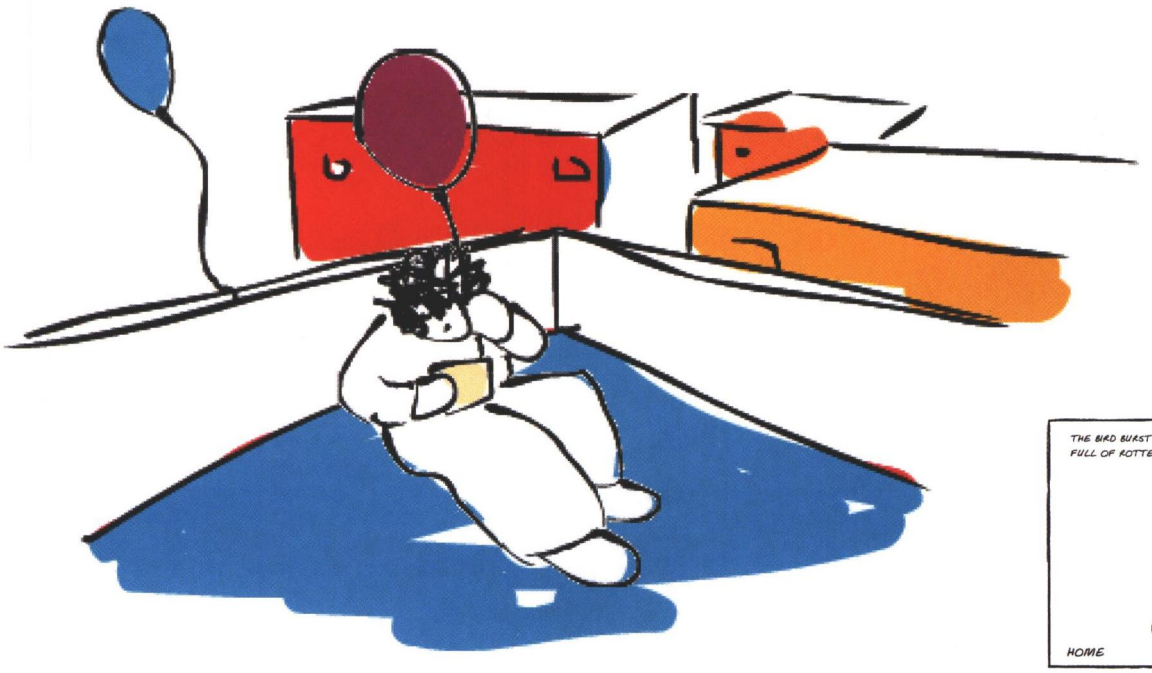
But there could be only one winner, and Tim's Balloon Sender took the award ahead of the online audio-visual experience Stephen had designed for Purr Records. Just over a year on, the two have continued collaborating and after freelancing together on various projects decided to form Float Design.

Balloon Sender, as well as the TRUMP TOWN site which we show you how to create in this issue of *Computer Arts Projects*, typify the kind of experiences Float aims to create from its base in Bath. By combining *Flash* with the dynamic databasing powers of PHP, sites can be richer and ever-changing.

"*Flash* is great as it is – you can do some fantastic, whizzy things," says Tim. "But it all becomes quite static after a while. There are a lot of sites out there that look beautiful, but they don't ever go much deeper than that. I think that's the interesting thing about putting in the PHP, which allows people to become a greater user, or to use the site differently."

Air mail

At Balloon Sender, you can send a balloon with a message on it into the virtual sky of PHP, then try and catch someone else's balloon, and then perhaps choose to contact them and respond to their random message via email. It's been suggested that this model



Balloon Sender, with its sketchy graphics, enables you to inflate a balloon, attach a text message and email contact to it, and let it loose into a virtual sky (or PHP database to you and me). Then you can try and catch a balloon in a game on the site, receive a message from someone else, and respond to them if the message interests you. [w] www.balloonsender.co.uk

Stephen Lockhart's Purr Records site was a nominated finalist the year Balloon Sender won the Mando Award for Tim. It's a punkrock feast of audio and video, and Stephen continues to make films with the record label. "One of the things which interests me about using the Web as a different medium is bringing the video aspect more into things as well," he says. [w] www.purr.org.uk/interactive

could be used in a meaningful way by charity sites. The user sponsors a balloon via micropayment, chooses their charity, and later hopefully receives a message back from another charitable donor who has caught their balloon.

"That's the aspect that we try and push – just trying to make Websites that are a little bit more interactive and more personal, I suppose," continues Tim. "It's quite voyeuristic in a way. A lot of it is looking at what other people have written, and that kind of gives a compelling aspect."

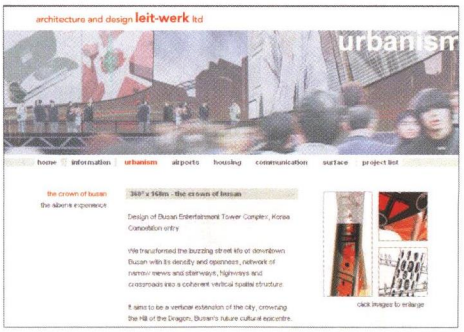
Mixed community

The TRUMP TOWN site adds to this with a community aspect that Float is looking to develop with more intricacy, again powered by PHP. "As the community grows, you've got different types of people so you sort of create subcultures within this town. And because it's trump cards, you've then got the opportunity of having actual packs of cards – mod cards, rockers, the Friday night girls going out," says Stephen.

While Tim and Stephen are developing ideas on the *Flash* and PHP front, a healthy amount of client work keeps them happily busy. They're signed up until 2006 working on a project for Cendia entitled TheMusicKey. Designed in *Flash*, this is a series of CD-ROM-based products to help teachers explain to pupils how to use music application *Logic*. One disc has been produced, with 11 more to follow, and the demo for the program has been used by Apple on its stand at computer shows. In addition, Float has done the branding, and is doing the print design.

Other client work is taking in everything from information-based sites for Bath Spa University, a project for social services in the London Borough of Merton, and a keepy-uppy football game to be used in viral marketing for Barry Vennison's Bid for Sport Website. They've also recently redesigned the site for the architects leit-werk. "We've been totally lucky in that we've had really good contacts who we've done work for and then they've passed us on to someone else," comments Tim.

Going forward, Float really wants to push in the direction of creating sites with dynamic content and communities, perhaps applying concepts from Tim and Stephen's fine art backgrounds as well as video. "We've got crazy ideas about how to develop stuff," says Tim. "We want to make it a lot more personal so we can add different things, different characters and maybe get different scores and that kind of thing." ca-p



The site created for the architects at leit-werk is a highlight amongst Float's client work. [w] www.leit-werk.com

You can contact Float Design at **[t]** +44 (0) 1225 333225, **[e]** hello@floatdesign.net and see more of the company's work at **[w]** www.floatdesign.net.

ENJOY THE COMMUNITY!
10M bit/s
10M bit/s
10M bit/s
HI
kak!
HI
boo
Illustration by Tado [w] www.tado.com

Build an online community

Using PHP to connect *Flash* to a MySQL database, create a site that enables users to add their own character to a growing community...

Most cheap Web-hosting packages now come equipped with PHP and a MySQL database, so it's easy to start creating Websites offering that bit extra in terms of interactivity. And now with the latest versions of *Flash*, things are made even easier when trying to add and retrieve information from a database.

This tutorial shows you how to create a version of Float Design's TRUMP TOWN website (http://www.floatdesign.net/trumptown). We show you how to create a trump character, gain skill and knowledge ratings, submit that card to a growing community of trump cards, and then print out and play. We have simplified the process, and pulled out the key elements to help you to create a *Flash* site that enables users to create their own character and add it to a growing community of cards.

We'll introduce you to the LoadVars object and the attachMovie method, show you how to construct your PHP pages, and talk you through the process of setting up your own database. As there are a number of steps involved in this process, we are assuming a reasonable knowledge of *Flash*. Don't be put off by this, because all the source files and scripts are also on the cover CD.

Expertise provided by Tim Ketterer and Stephen Lockhart of Float Design. [w] www.floatdesign.net.

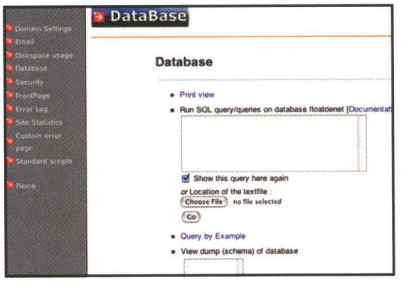
All the components needed to complete this tutorial can be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Flash and PHP folder.

Part 1: The database

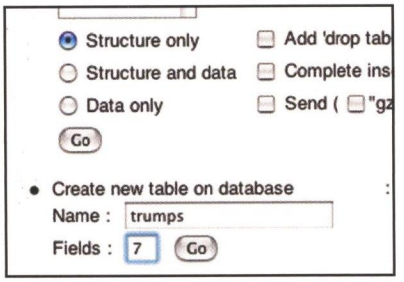
Setting up your MySQL database with PHPMyAdmin...

Local software

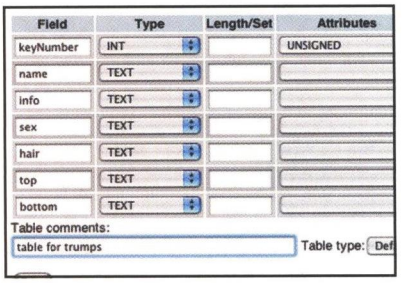
An alternative to buying a hosting package would be to run a MySQL database and PHP locally on your machine by installing the appropriate software.



1 For this tutorial we bought a cheap hosting package, which included PHP4 and one MySQL database. The advantage of buying this service is that everything is installed for you, plus they offer PHPMyAdmin, which makes setting up your database a relative breeze.



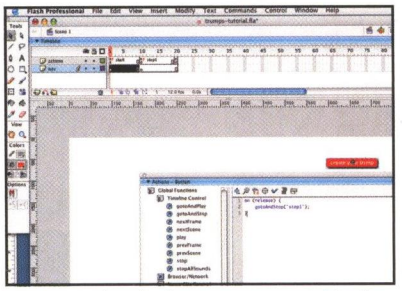
2 Using PHPMyAdmin, create a new table on your database. In the Name field enter trumps and in the Fields field enter 7. Then press Go.



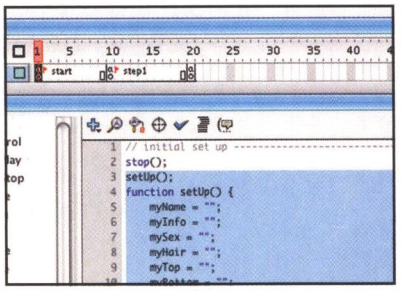
3 Set up the Fields as above, making sure that the keyNumber field has Auto–increment selected in the Extra tab as well as Primary being selected. Check the values for Type, making sure that all fields correlate to their related 'type' as per the screenshot. When this is done, press Save.

Part 2: Setting up in Flash

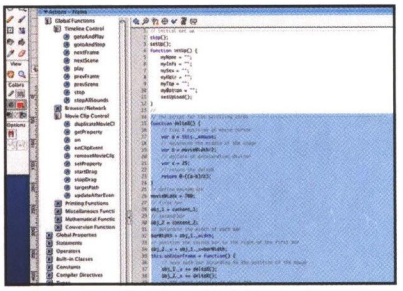
Prepare the timeline and the initial ActionScript...



1 Create three keyframes on your main Timeline, 10 frames apart. Create an actions layer, and put a stop() in each frame. Give each a frame label named: start, step1, and finish. Create a button on the stage for creating a trump, and within an on (release) statement set it to gotoAndStop ("step1");



2 On the first frame of the main timeline create a function called setUp(). This initialises the variables to hold the info that will be sent to the PHP. Call these: mySex, myName, myInfo, myHair, myTop, myBottom, and give them a default value of "" (nothing). Also trigger the setUpLoad() function.



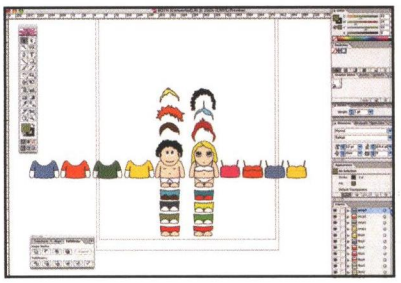
3 We've used a piece of script to create a scrolling clip of these 18 random trump cards, which moves across the bottom of the screen in accordance with the user's mouse. This isn't necessary to the functionality of the site, but the full code is within scrollingClip.txt on the cover CD.

Part 3: Creating the elements

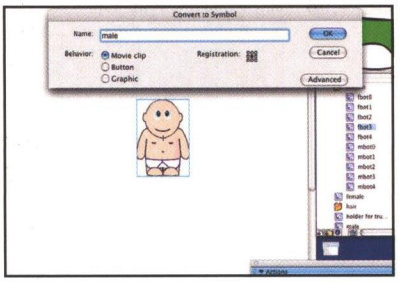
Design your character and prepare its properties...

Vector elements

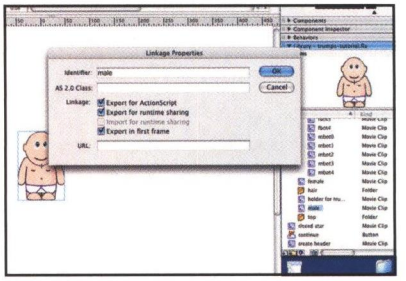
We used *Illustrator* to create these elements, though you could just as easily use *Flash*. Don't use a bitmap program such as *Photoshop*, because the elements need to be vectors so they can be scaled down later.



1 To create the various elements of each trump, create 8 hairstyles (4 each for male and female), 8 tops (again, 4 x 2), and 8 bottoms. You will also need a male and female body, and a male and female background for the cards.



2 Put these separate elements in their own MovieClips. Call the two bodies: male, female; the backgrounds: bgmale, bgfemale; the hair/top/bottom clips: mhair1, mhair2, mtop1 for male and fbot1, fhair2, etc. for female. We also created empty MCs called mhair0, ftop0 for the bald/naked look.



3 For each of these MovieClips, Ctrl-click (right-click) on them in the library, and choose Linkage. Check the option 'Export for ActionScript'. The two background MCs also contain two dynamic text boxes with the variables: nameIn, and info. These will output the name and information.

The LoadVars object

The LoadVars object is used to send and receive variables from *Flash* to PHP and vice versa. The object acts as a holder for these variables. If you want to send a variable to PHP from *Flash*, you can do it by creating a new loadVars object:

```
myLoadVars = new LoadVars();
```

Variables to be sent to PHP within the LoadVars object should be referred to as:

```
myLoadVars.variableName;
```

To hold received data from PHP a new LoadVars is created:

```
ReplyData = new LoadVars();
```

The function myOnLoad runs once ReplyData has received the variables back from PHP:

```
ReplyData.onLoad = myOnLoad;
```

MyOnLoad function determines the result of replyData. Variables that are sent from PHP to *Flash* are referred to by using 'this' – i.e. 'this.variable' – as they are contained within the replyData object:

```
myVar = this.myVar;
```

We send the variables to PHP by appending the sendAndLoad() function to the new LoadVars object. We then add the following parameters to the sendAndLoad function:

```
myLoadVars.sendAndLoad(Name of php file, the
receiving LoadVars object, and the method in
which you are sending the data - i.e. GET
or POST )
```

Part 4: Using loadVars

Set up the various LoadVars objects to talk to PHP...

```
16 function setUpLoad() {
17     formData = new LoadVars();
18     formData.cardMade = _root.cardMade;
19     //
20     replyData = new LoadVars();
21     replyData.onLoad = myOnLoad;
22     //
23     runPhp();
24 }
25 function runPhp() {
26     formData.sendAndLoad("retrieve.php", r
27 }
28 function myOnLoad(success) {
29     if (success) {
```

1 In frame 1 of the actions layer on the _root level you need to set up a function called setUpLoad(), initiated by the setUp() function.

```
16 function setUpLoad() {
17     formData = new LoadVars();
18     formData.cardMade = _root.cardMade;
19     //
20     replyData = new LoadVars();
21     replyData.onLoad = myOnLoad;
22     //
23     runPhp();
24 }
25 function runPhp() {
26     formData.sendAndLoad("retrieve.php", replyData,
27 }
28 function myOnLoad(success) {
29     if (success) {
30         //
31         _root.cardMade = "";
```

2 In this function you need to set up two new LoadVars objects – formData to send the data and replyData to receive it. Set up the variable cardMade within formData, which will be sent to retrieve.php. We will refer to cardMade later in the tutorial.

```
14 //
15 // script to set up the load vars for sending and recieving var
16 function setUpLoad() {
17     formData = new LoadVars();
18     formData.cardMade = _root.cardMade;
19     //
20     replyData = new LoadVars();
21     replyData.onLoad = myOnLoad;
22     //
23     runPhp();
24 }
25 function runPhp() {
26     formData.sendAndLoad("retrieve.php", replyData, "post");
27 }
28 function myOnLoad(success) {
29     if (success) {
30         //
31         _root.cardMade = "";
32         //
33         // set up main card
```

3 Next, make functions runPhp() to send the data and myOnLoad(), both are initiated by setUpLoad(). In myOnLoad() you need to set _root.cardMade to="" (nothing) as well as setting the values of all the *Flash* variables associated with the cards to the values received by PHP. (See Steps 2-6 of Part 5 of this tutorial.)

```
410     maincard_mc.holder_mc.bg_mc.score2 = card5;
411     maincard_mc.holder_mc.attachMovie(card1, "sex_mc
412     maincard_mc.holder_mc.attachMovie(card8, "bot_mc
413     maincard_mc.holder_mc.attachMovie(card7, "top_mc
414     maincard_mc.holder_mc.attachMovie(card6, "hair_m
415 }
416 //
417 // the functions for sending all the info between fl
418 // submit card function
419 function submitCard() {
420     cardData = new LoadVars();
421     cardReplyData = new LoadVars();
422     cardReplyData.onLoad = cardOnLoad;
423     //
424     runCardPhp();
425 }
426 function runCardPhp() {
```

4 Next, you must set up function submitCard() at frame 1 of the _root timeline. Again, two LoadVars objects need to be initiated. cardData is used to send variables and cardReplyData is used to receive data.

```
402 // submit card function
403 function submitCard() {
404     cardData = new LoadVars();
405     //
406     cardData.myName = _root.myName;
407     cardData.myInfo = _root.myInfo;
408     cardData.mySex = _root.mySex;
409     cardData.myHair = _root.myHair;
410     cardData.myTop = _root.myTop;
411     cardData.myBottom = _root.myBottom;
412     //
413     cardReplyData = new LoadVars();
414     cardReplyData.onLoad = cardOnLoad;
415     //
```

5 Within cardData you must send all the variable values associated with the new card to PHP. These can then be entered into the database.

```
407     cardData.myInfo = _root.myInfo;
408     cardData.mySex = _root.mySex;
409     cardData.myHair = _root.myHair;
410     cardData.myTop = _root.myTop;
411     cardData.myBottom = _root.myBottom;
412     //
413     cardReplyData = new LoadVars();
414     cardReplyData.onLoad = cardOnLoad;
415     //
416     runCardPhp();
417 }
418 function runCardPhp() {
419     cardData.sendAndLoad("details.php", cardReplyData, "post");
420 }
421 function cardOnLoad(success) {
422     if (success) {
423         _root.gotoAndStop("start");
424     }
425 }
426
```

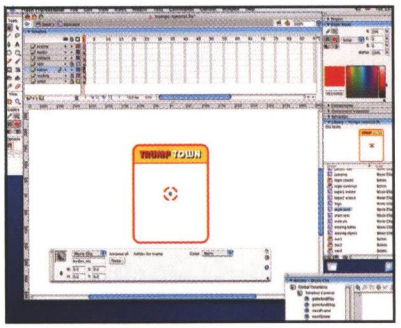
6 Create two more functions within the actions layer, to run from submitCard(). runCardPhp() sends *Flash's* variables to PHP, and cardOnLoad() sends the *Flash* file back to the start of the _root level and runs the functions in the first frame.

Part 5: How the cards work

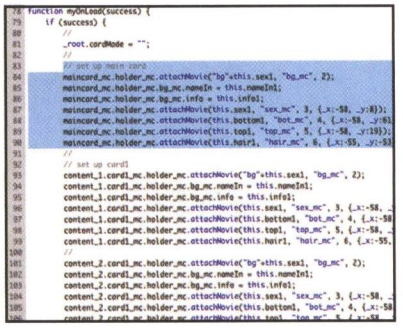
Building the trump card MovieClips and using attachMovie…

The same script

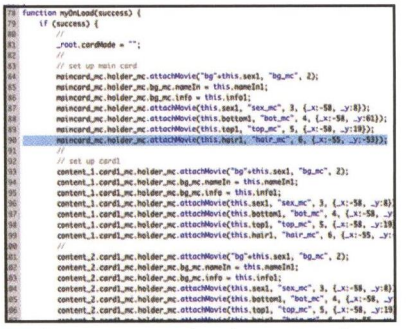
Don't be put off by the amount of script used in this section, because it's merely seven lines duplicated 32 times with a minute change each time.



1 Create a new MovieClip called main card, in which you'll create a loader animation, logo, and border for the card. On a separate layer, create an empty MovieClip with an instance name of holder_mc. This is where you'll attach the various elements of each trump card using the attachMovie() method.

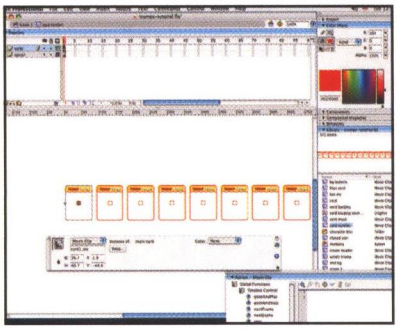


2 Bring out an instance of 'main card' onto the stage at the first frame, and give it an instance name of maincard_mc. The target path to this card for the attachMovie() method is: maincard_mc.holder_mc.attachMovie, because you'll be putting these MovieClips into the holder clip you created earlier.

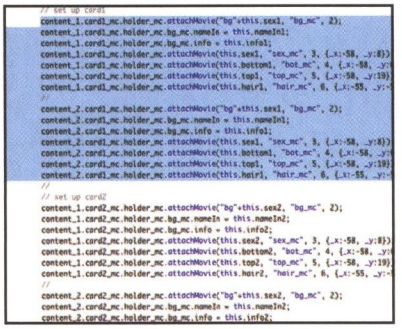


3 You can pass a number of parameters when using attachMovie():

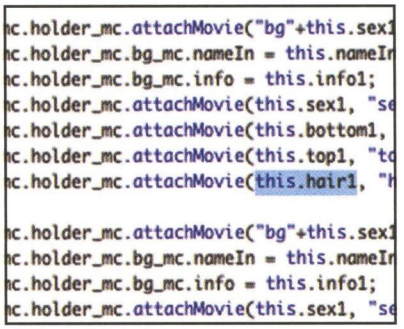
- The MovieClip you want to attach (this will be the linkage name you set).
- An instance name.
- A level onto which it will sit (each MovieClip must be on a different level).
- An initObject with x and y co-ordinates.



4 To create the MovieClip with the 18 small cards, create a new clip called card scroller. Inside this, drag out 18 instances of the main card MovieClip and scale them down in size so seven fit on the stage at once. Give them instance names of card1_mc through to card18_mc.



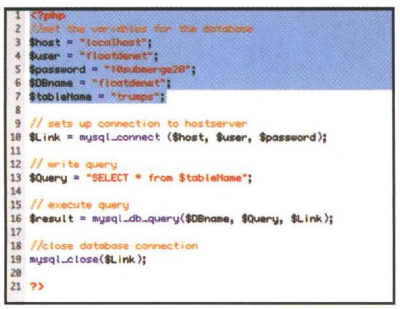
5 Make two duplicate instances of the card scroller clip, giving them instance names of content_1 and content_2. The target paths for these attachMovie() methods need to be assigned twice. Duplicate this script 18 times, for the 18 cards.



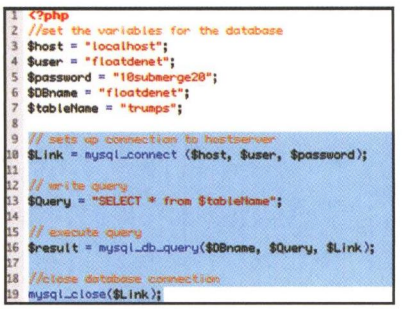
6 As these methods are within a function that's triggered through the replyData object by PHP passing the variables to *Flash*, we refer to these variables that set the hair, sex, top, and so on, using 'this' – i.e. this.hair1.

Part 6: Creating the PHP

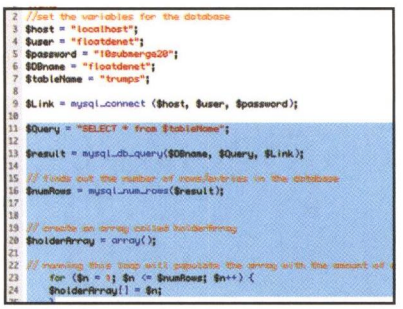
Constructing PHP pages to enable *Flash* to talk to the database…



1 Every PHP page that accesses a database is set up in the same way: open and close PHP tags surround the code. Variables are set up for the hostname, username, password, database name and table name, so they can be referred to easily.



2 Connection with the host server is established using the msql_connect function. Database Queries are executed using mysql_db_query(). When queries have been executed, the connection to the database must be closed using mysql_close().



3 The first PHP script we run is to populate the 18 cards with the variables from the database. First, run a query to find out how many entries are in the database. Create a new array and populate this with the amount of elements in the database.

PHP script

Unlike *Flash*, PHP uses a '$' in front of the string name when setting up variables. For example: $myVar="hello";

You also need to make sure you use the semi-colon where necessary otherwise PHP will throw up an error and the script won't work.

```
$result = mysql_db_query($DBname, $Query, $Link);

// finds out the number of rows/entries in the data
$numRows = mysql_num_rows($result);


// create an array called holderArray
$holderArray = array();

// running this loop will populate the array with t
    for ($n = 1; $n <= $numRows; $n++) {
    $holderArray[] = $n;
    }


// if the card has just been made then run this if
if($cardMade=="made"){
    // takes the last element out of the array
    $lastOne = array_pop ($holderArray);
    // randomly shuffles the contents of the array
    shuffle($holderArray);
```

4 Now create an 'if' statement that checks to see if the variable cardMade ="made". This variable is passed from *Flash* when a new card has just been made, because we want this new card to eventually appear first in the array so that it populates the main card.

```
$result = mysql_db_query($DBname, $Query, $Link);

// finds out the number of rows/entries in the database
$numRows = mysql_num_rows($result);


// create an array called holderArray
$holderArray = array();

// running this loop will populate the array with the amount of entries in
    for ($n = 1; $n <= $numRows; $n++) {
    $holderArray[] = $n;
    }

// if the card has just been made then run this if statement to send it to
if($cardMade=="made"){
    // takes the last element out of the array
    $lastOne = array_pop ($holderArray);
    // randomly shuffles the contents of the array
    shuffle($holderArray);
    // puts the $numrows variable (which was the last element) to the front
    array_unshift($holderArray, $numRows);
} else {
    // randomly shuffles the contents of the array
    shuffle($holderArray);
}
```

5 This is done by removing this last element from the array, shuffling the array to create a random order, and then putting this element into the first position of the array. If the variable cardMade ="no" we just simply shuffle the array.

```
// if the card has just been made then run this if
if($cardMade=="made"){
    // takes the last element out of the array
    $lastOne = array_pop ($holderArray);
    // randomly shuffles the contents of the array
    shuffle($holderArray);
    // puts the $numrows variable (which was the l
    array_unshift($holderArray, $numRows);
} else {
    // randomly shuffles the contents of the array
    shuffle($holderArray);
}

$count = 1;
// set this condition to be the number of cards th
while ($count <= 18) {

    $count++;
```

6 In order to take the variables of all 18 cards out of the database, we need to set up a while loop that runs 18 times. This is done by using a variable called $count, which increments by one each time it loops until $count =18.

```
// running this loop will populate the array with the amount of entr
    for ($n = 1; $n <= $numRows; $n++) {
    $holderArray[] = $n;
    }


// if the card has just been made then run this if statement to send
if($cardMade=="made"){
    // takes the last element out of the array
    $lastOne = array_pop ($holderArray);
    // randomly shuffles the contents of the array
    shuffle($holderArray);
    // puts the $numrows variable (which was the last element) to the
    array_unshift($holderArray, $numRows);
} else {
    // randomly shuffles the contents of the array
    shuffle($holderArray);
}

$count = 1;
// set this condition to be the number of cards that you want to bri
while ($count <= 18) {
    $randomNum = $holderArray[0];
    $Query2 = "SELECT * from $tableName where(keyNumber='$randomNum'
    $result2 = mysql_db_query($DBname, $Query2, $Link);
    $row = mysql_fetch_array($result2);
```

7 As each loop of the script runs, it picks out the number being held in the first position of the array – $holderArray[0]. Create a variable $randomNumber which is populated by this number from the array, and use this to retrieve that element from the database using the query: where keyNumber ='$randomNumber'.

```
    // puts the $numRows variable (which was the last ele
    array_unshift($holderArray, $numRows);
} else {
    // randomly shuffles the contents of the array
    shuffle($holderArray);
}

$count = 1;
// set this condition to be the number of cards that you
while ($count <= 18) {
    $randomNum = $holderArray[0];
    $Query2 = "SELECT * from $tableName where(keyNumber='
    $result2 = mysql_db_query($DBname, $Query2, $Link);

    $row = mysql_fetch_array($result2);

    // populate the variables
    echo ("&name1n".$count."=" . urlencode($row[name]));
    echo ("&sex".$count."=" . urlencode($row[sex]));
    echo ("&hair".$count."=" . urlencode($row[hair]));
    echo ("&top".$count."=" . urlencode($row[top]));
    echo ("&bottom".$count."=" . urlencode($row[bottom]));
```

8 In order to send these variables back to *Flash* you need to tell PHP to populate *Flash's* variables with the information gathered from each row within the database. This is done using urlencode. We also concatenate the variables with $count to correlate with the *Flash* variables that run from 1-18.

```
}

$count = 1;
// set this condition to be the number of cards
while ($count <= 18) {
    $randomNum = $holderArray[0];
    $Query2 = "SELECT * from $tableName where(k
    $result2 = mysql_db_query($DBname, $Query2,

    $row = mysql_fetch_array($result2);

    // populate the variables
    echo ("&name1n".$count."=" . urlencode($row
    echo ("&sex".$count."=" . urlencode($row[se
    echo ("&hair".$count."=" . urlencode($row[h
    echo ("&top".$count."=" . urlencode($row[to
    echo ("&bottom".$count."=" . urlencode($row
    echo ("&info".$count."=" . urlencode($row[i
```

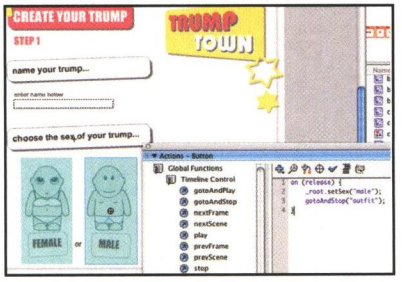
9 After all the variables are pulled out during this loop, the array_shift() function is used to remove what is in $holderArray[0] and shift everything down one. The next loop will still target $holderArray[0], but its contents will have changed from the previous time the loop ran.

Part 7: Character building

Piece together the elements to create the dynamic character...

```
// functions to set up the trump on the main card
function setName() {
    myCard_mc.holder_mc.bg_mc.nameIn = myName;
}
//
function setSex(whichSex) {
    mySex = whichSex;
    myCard_mc.holder_mc.attachMovie("bg"+mySex, "bg_mc", 2);
    myCard_mc.holder_mc.attachMovie(mySex, "sex_mc", 3, {_x:-58, _y:8});
    // set the holder variables to a default value
    if (whichSex == "male") {
        myHair = "mhair0";
        myTop = "mtop0";
        myBottom = "mbot0";
    } else {
        myHair = "fhair0";
        myTop = "ftop0";
        myBottom = "fbot0";
    }
    // run the set name function
    setName();
}
//
function setHair(whichHair) {
    myCard_mc.holder_mc.attachMovie(whichHair, "hair_mc", 6, {_x:-55, _y:-53});
    myHair = whichHair;
```

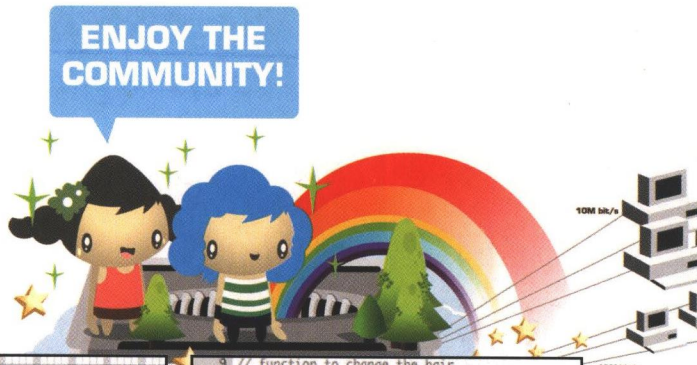
1 On the frame labelled step1 of the main timeline, in the actions layer create functions assigning values to the character's variables: setName(), setSex(), setHair(), setTop(), setBottom(). Create a MovieClip on the timeline, containing two separate keyframes labelled: sex, and outfit.



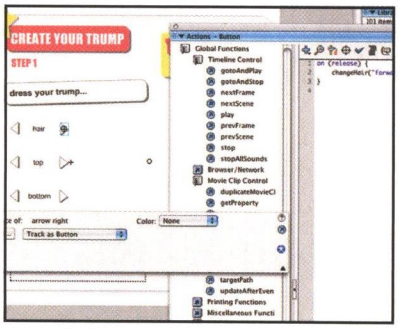
2 At the frame labelled sex, create an input textbox with the variable: _root.myName. Drag out an instance of the male and female characters and create invisible buttons over each. Within on(release) statements run the function setSex(), passing the parameter of male or female.

```
// functions to set up the trump on the main card
function setName() {
    myCard_mc.holder_mc.bg_mc.nameIn = myName;
}
//
function setSex(whichSex) {
    mySex = whichSex;
    myCard_mc.holder_mc.attachMovie("bg"+mySex, "bg_mc", 2);
    myCard_mc.holder_mc.attachMovie(mySex, "sex_mc", 3, {_x:-58, _y:8});
    // set the holder variables to a default value
    if (whichSex == "male") {
        myHair = "mhair0";
        myTop = "mtop0";
        myBottom = "mbot0";
    } else {
        myHair = "fhair0";
        myTop = "ftop0";
        myBottom = "fbot0";
    }
    // run the set name function
    setName();
}
```

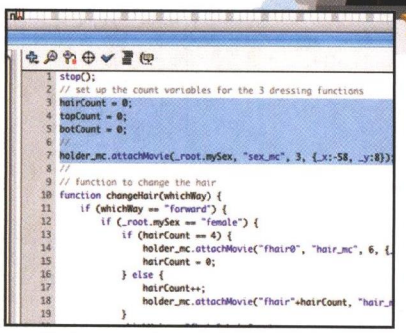
3 This _root.setSex() function sets the holder variable for this character's sex – mySex – to whichever parameter was passed. It attaches the background clip and body to the card and sets the variables for myHair, myTop, myBottom, and runs setName() passing the myName variable to the card.



Character building continued...

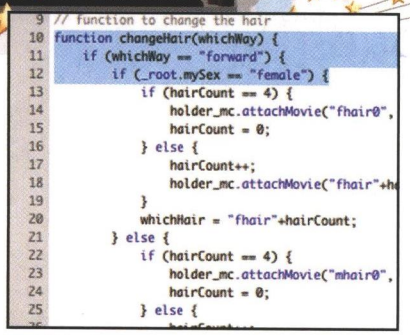


4 In the frame 'outfit' create two arrow clips and drag out three instances of each. They'll run the various functions on the actions layer, passing a parameter of forward or back depending which arrow is pressed. Create an empty MovieClip on the stage with an instance name of holder_mc.



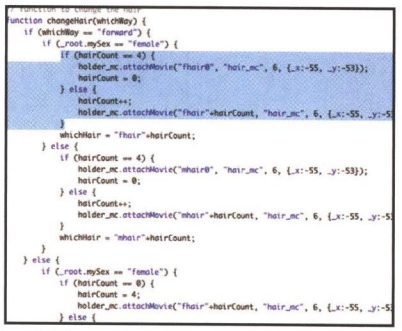
```
stop();
// set up the count variables for the 3 dressing functions
hairCount = 0;
topCount = 0;
botCount = 0;
//
holder_mc.attachMovie(_root.mySex, "sex_mc", 3, {_x:-58, _y:8});
//
// function to change the hair
function changeHair(whichWay) {
    if (whichWay == "forward") {
        if (_root.mySex == "female") {
            if (hairCount == 4) {
                holder_mc.attachMovie("fhair0", "hair_mc", 6, {
                hairCount = 0;
            } else {
                hairCount++;
                holder_mc.attachMovie("fhair"+hairCount, "hair_
            }
```

5 On the actions layer create three variables – hairCount, topCount, and botCount – and set to 0. Using attachMovie() attach the trump's body MovieClip into the holder MovieClip. The variable _root.mySex will tell us which clip to attach.



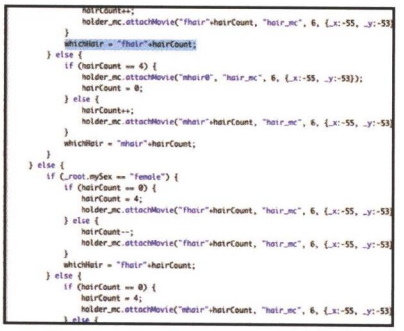
```
// function to change the hair
function changeHair(whichWay) {
    if (whichWay == "forward") {
        if (_root.mySex == "female") {
            if (hairCount == 4) {
                holder_mc.attachMovie("fhair0",
                hairCount = 0;
            } else {
                hairCount++;
                holder_mc.attachMovie("fhair"+h
            }
            whichHair = "fhair"+hairCount;
        } else {
            if (hairCount == 4) {
                holder_mc.attachMovie("mhair0",
                hairCount = 0;
            } else {
```

6 Using hair as an example, the changeHair() function on the actions layer first checks to see if the parameter is forward/back, then checks the main mySex var to see if it's male/female. As you have created five options of hair style (including a no-hair option), this gives five possible outcomes.



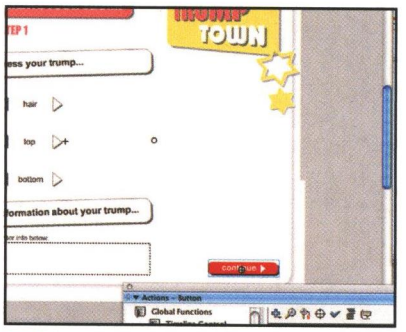
```
function changeHair(whichWay) {
    if (whichWay == "forward") {
        if (_root.mySex == "female") {
            if (hairCount == 4) {
                holder_mc.attachMovie("fhair0", "hair_mc", 6, {_x:-55, _y:-53});
                hairCount = 0;
            } else {
                hairCount++;
                holder_mc.attachMovie("fhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-5
            }
            whichHair = "fhair"+hairCount;
        } else {
            if (hairCount == 4) {
                holder_mc.attachMovie("mhair0", "hair_mc", 6, {_x:-55, _y:-53});
                hairCount = 0;
            } else {
                hairCount++;
                holder_mc.attachMovie("mhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-5
            }
            whichHair = "mhair"+hairCount;
        }
    } else {
        if (_root.mySex == "female") {
            if (hairCount == 0) {
                hairCount = 4;
                holder_mc.attachMovie("fhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-5
            } else {
```

7 Each time the function is run, it increments the count variable by one, then uses attachMovie to attach the correct MovieClip – that is, "fhair"+hairCount. Once the count reaches four, it's then sent back to 0 to stop it continuing. If the passed parameter is 'back', it reduces the count variable by one rather than increasing it.



```
            holder_mc.attachMovie("fhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-53});
        }
        whichHair = "fhair"+hairCount;
    } else {
        if (hairCount == 4) {
            holder_mc.attachMovie("mhair0", "hair_mc", 6, {_x:-55, _y:-53});
            hairCount = 0;
        } else {
            hairCount++;
            holder_mc.attachMovie("mhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-53
        }
        whichHair = "mhair"+hairCount;
    }
} else {
    if (_root.mySex == "female") {
        if (hairCount == 0) {
            hairCount = 4;
            holder_mc.attachMovie("fhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-53
        } else {
            hairCount--;
            holder_mc.attachMovie("fhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-53
        }
        whichHair = "fhair"+hairCount;
    } else {
        if (hairCount == 0) {
            hairCount = 4;
            holder_mc.attachMovie("mhair"+hairCount, "hair_mc", 6, {_x:-55, _y:-53
```

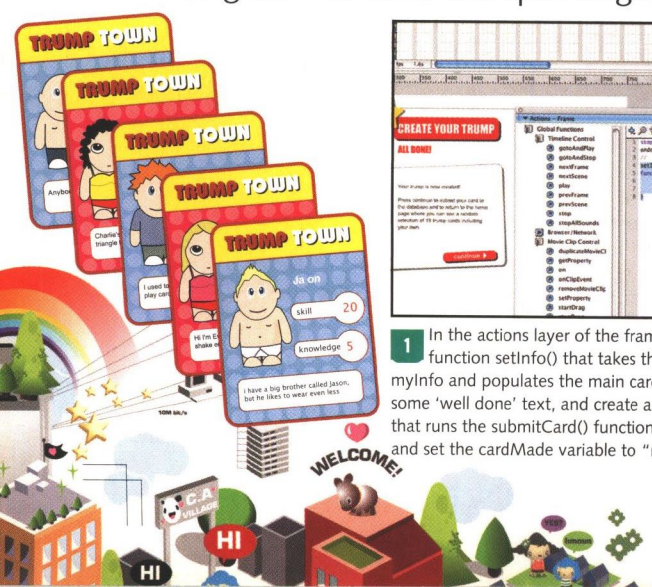
8 Then create a variable called whichHair, to which you then assign the current hair clip – whichHair = "fhair"+hairCount. At the end of this function, run setHair() on the main timeline, passing this parameter of whichHair, which attaches the correct hair m/c to the main card and sets the main myHair variable to the whichHair value.

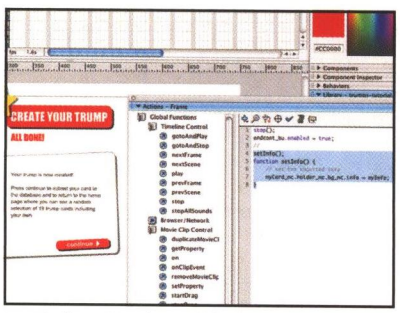


9 To receive the trump's information, create an input text field with the variable _root.myInfo. Also create a continue button that when pressed will send the user onto the frame labelled finish, on the main timeline.

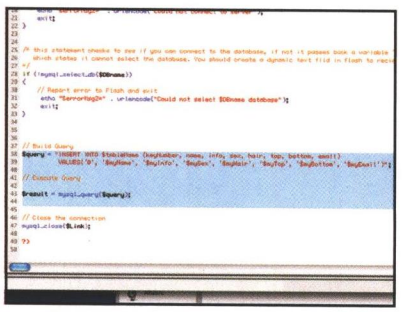
Part 8: Finishing up

Doing the final tasks and uploading...





1 In the actions layer of the frame 'finish', run the function setInfo() that takes the info variable myInfo and populates the main card with this. Write some 'well done' text, and create a continue button that runs the submitCard() function at frame 'start', and set the cardMade variable to "made".



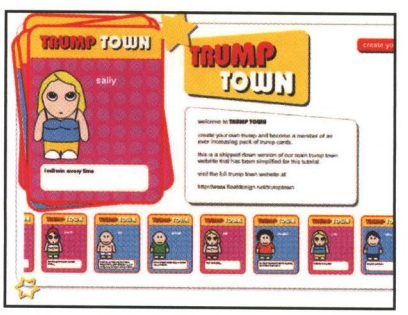
```
exit();
}
// this statement checks to see if you can connect to the database, if not it passes back a variable
// which states it cannot select the database. You should create a dynamic text field in flash to reca
//
if (!mysql_select_db($DBname))
{
    // Report error to Flash and quit
    echo "&errorMsg2=" . urlencode("Could not select $DBname database");
    exit();
}

// Build Query
$query = "INSERT INTO $tablename (keyNumber, names, info, eye, hair, top, bottom, email)
        VALUES('0', '$myName', '$myInfo', '$myEye', '$myHair', '$myTop', '$myBottom', '$myEmail')";

// Execute Query
$result = mysql_query($query);

// Close the connection
mysql_close($DB_link);
?>
```

2 This is the final PHP script you need to create, which takes all the variables loaded into the cardData loadVars object (that is, all the variables that make up a new card) and enters them into the database. The variables are entered using 'INSERT' in the query.

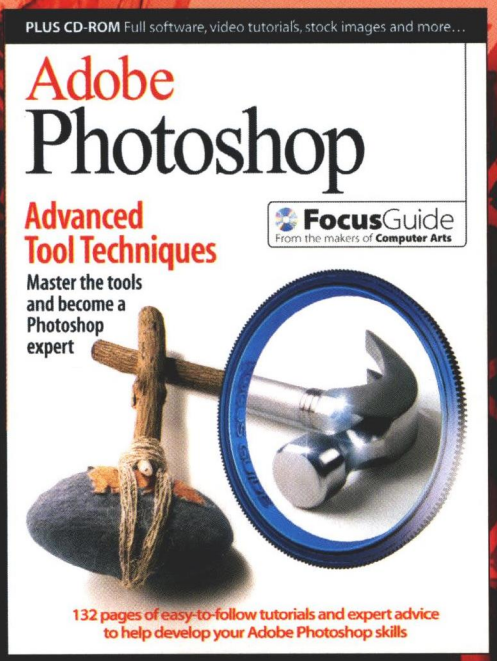


3 Now upload your files to your server, making sure you put all the files in the same folder. You should now be able to play! See a working version of TRUMP TOWN at: www.floatdesign.net/trumptown/tutorial. **ca-p**

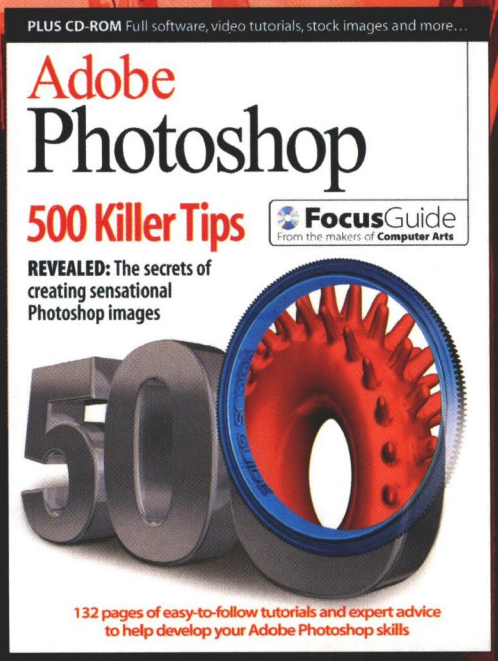
Adobe Photoshop

FocusGuide
From the makers of Computer Arts

132 INFORMATION-PACKED PAGES PLUS FREE CD-ROM


PLUS CD-ROM Full software, video tutorials, stock images and more...
Adobe
Photoshop
Advanced Tool Techniques
FocusGuide
From the makers of Computer Arts
Master the tools and become a Photoshop expert
132 pages of easy-to-follow tutorials and expert advice to help develop your Adobe Photoshop skills

ON SALE 29 JULY


PLUS CD-ROM Full software, video tutorials, stock images and more...
Adobe
Photoshop
500 Killer Tips
FocusGuide
From the makers of Computer Arts
REVEALED: The secrets of creating sensational Photoshop images
132 pages of easy-to-follow tutorials and expert advice to help develop your Adobe Photoshop skills

ON SALE 26 AUGUST

The world's best image editing program isn't just for the pros! We'll help you master Photoshop's essential tools and features

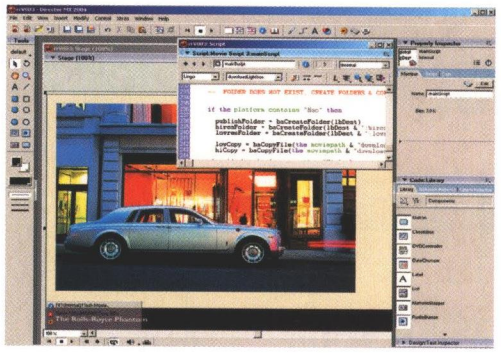
GET YOUR COPY NOW

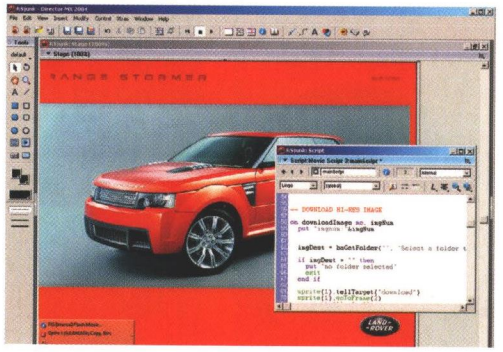
Available from WHSmith and other retailers

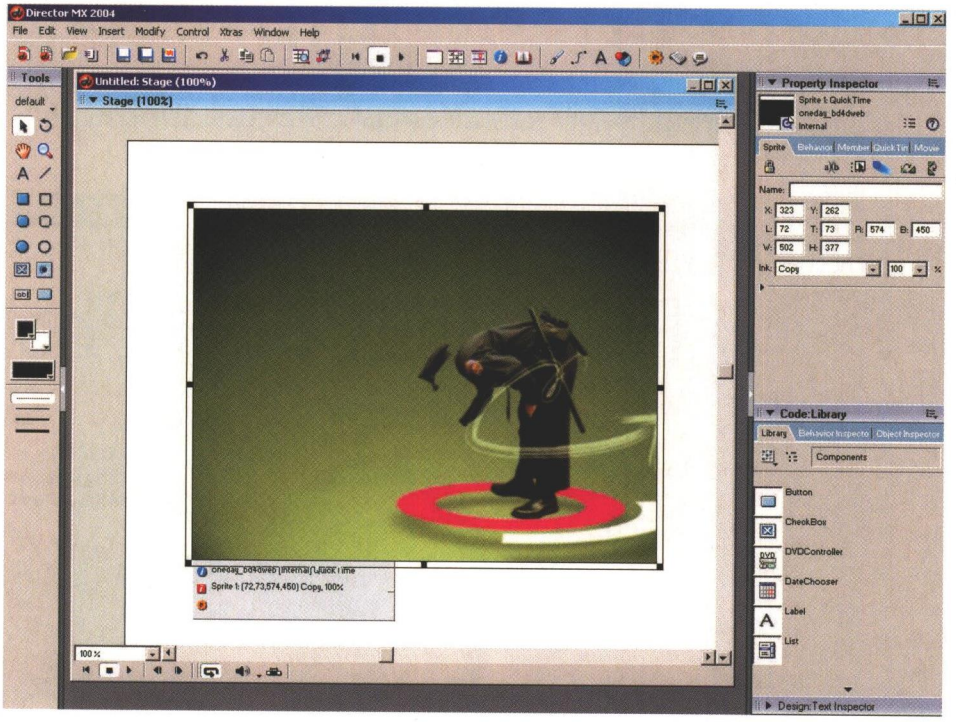
CONTENTS SUBJECT TO CHANGE

Win Director and Flash!

Win the ultimate multimedia bundle from Macromedia worth over £1500...







We thought it only right this month to offer you the chance to win the ultimate multimedia bundle – a copy of both *Flash MX 2004 Professional* and *Director MX 2004*. Using these two industry-standard applications together you can create stunning, engaging multimedia for the Web, CD-ROM or DVD. This is a bundle worth over £1500 and is essential if you want to create cutting-edge multimedia – so enter now!

Flash MX 2004 Professional

Flash MX 2004 Professional is the ultimate tool for creating dynamic Web and CD-ROM-based animations and Web applications. This new version of the industry-standard tool includes several new features to make building animations and complex Web applications a more enjoyable, and profitable, experience.

Timeline effects are a great new feature for those using *Flash* as an animation tool, offering easy-to-use, automated animated effects. EPS and PDF file support makes it easier than ever to make more appealing *Flash* movies, and new accessibility and globalisation tools make it simpler to cater for a wider audience.

Forms are also a new feature, giving you the ability to create forms-based applications in a similar manner to such tools as Microsoft Visual Basic. The enhanced video features are great for anyone wanting to create low-bandwidth, yet visually striking, *Flash*-based video content, and ActionScript 2.0 provides an easier route in for those new to the language. Advanced components — making it easy to create normally complex elements — and mobile phone development tools round off the new features. Team up *Flash* with *Director* and you have a formidable software suite...

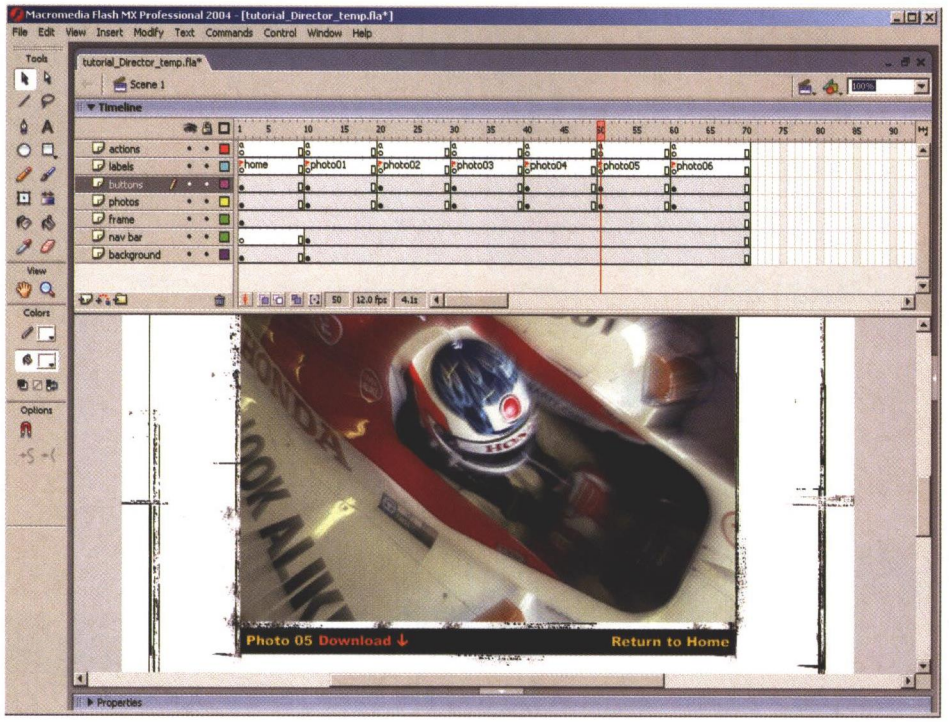
Director MX 2004

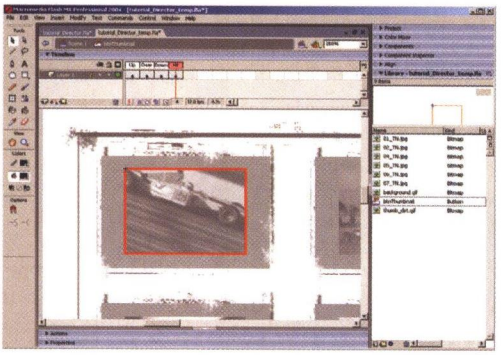
What can we say about *Director?* Well, it's over 20 years old and has become the de facto standard in multimedia authoring. Nothing else comes close to giving the kind of power you get from this application.

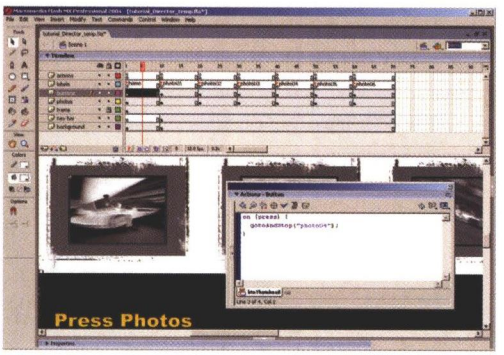
The new MX 2004 version of the application has a vast range of new features, all aimed at making it easier to create stunning multimedia. The first, and possibly most significant new feature, is the support for JavaScript syntax. This means experienced developers can mix JavaScript and Lingo to create highly developed applications, while newcomers to Lingo can find their feet with JavaScript before tackling *Director's* native programming language.

The next significant new feature is the ability to publish Mac and PC projector files from a single copy. The inclusion of a revamped

macromedia







Projector Publishing Panel makes this a cinch. DVD-video support is next, and you can play back DVD video and link events to Web pages – meaning you can explain what's going on as the DVD plays. See our tutorial on page 66 for an in-depth guide to using this feature.

There's also support for more video formats, including MPEG-4 video and MP3 audio, and Sprite and Channel naming, improved panel docking and much more.

You'd be mad to not want to get your hands on *Director MX 2004* and *Flash MX Professional* – and thanks to us and Macromedia, you can, for free!

The question

What is the name of *Director's* native programming language?

To enter

The easiest way to enter this competition is via our Website. Log onto **[w] www.computerarts.co uk/competition**. To enter, click the link at the bottom of the page and fill in the form, including your answer and contact details.

If you don't have Web access, you can enter by simply sending us your answer on a postcard to:
Macromedia competition, *Computer Arts Projects*, 30 Monmouth Street, Bath, BA1 2BW.

Please include your name, address and a telephone number so we can contact you if you win.

The rules

The closing date for entries is 22 August 2004. Employees of Macromedia and Future Publishing, their agents and families are not permitted to enter. Multiple entries are not accepted. The editor's decision is final and there are no cash alternatives. No other correspondence will be entered into. If you are entering by post and do not wish any of the companies involved in this competition to contact you with further offers, please indicate this on your entry. We will not pass your details on to third parties. ca-p

For more information on *Flash MX 2004 Professional* and *Director MX 2004*, visit **[w]** www.macromedia.com/software/

Illustration by Lasse Skarbovik. Contact via The Organisation [e] organise@easynet.co.uk

Building Flash components

Create fully customisable interface elements from scratch...

Over the following pages we show you how to build two components, combine them into a colour palette and address the colour properties of a MovieClip. The components will have all the standard features of professional components, including inspectable properties. The designs are based on the standard component template used by professional developers all over the world. Professional component design is based around the new class model introduced in *MX 2004*. This new style of development means that you can build beautifully crafted components that any designer would be proud to present.

The example we're building will be a bridge between design as traditionally done in *Flash* and the new class-based design method. One of the criticisms of the new model has been the file size, but we will build our component with file sizes that are similar, if not smaller, than old-style MovieClips. Let's start building...

Expertise provided by David Aaron. You can contact David at **[e]** david.aaron@thatplanet.net.

The files needed to complete this tutorial can be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Flash components folder.

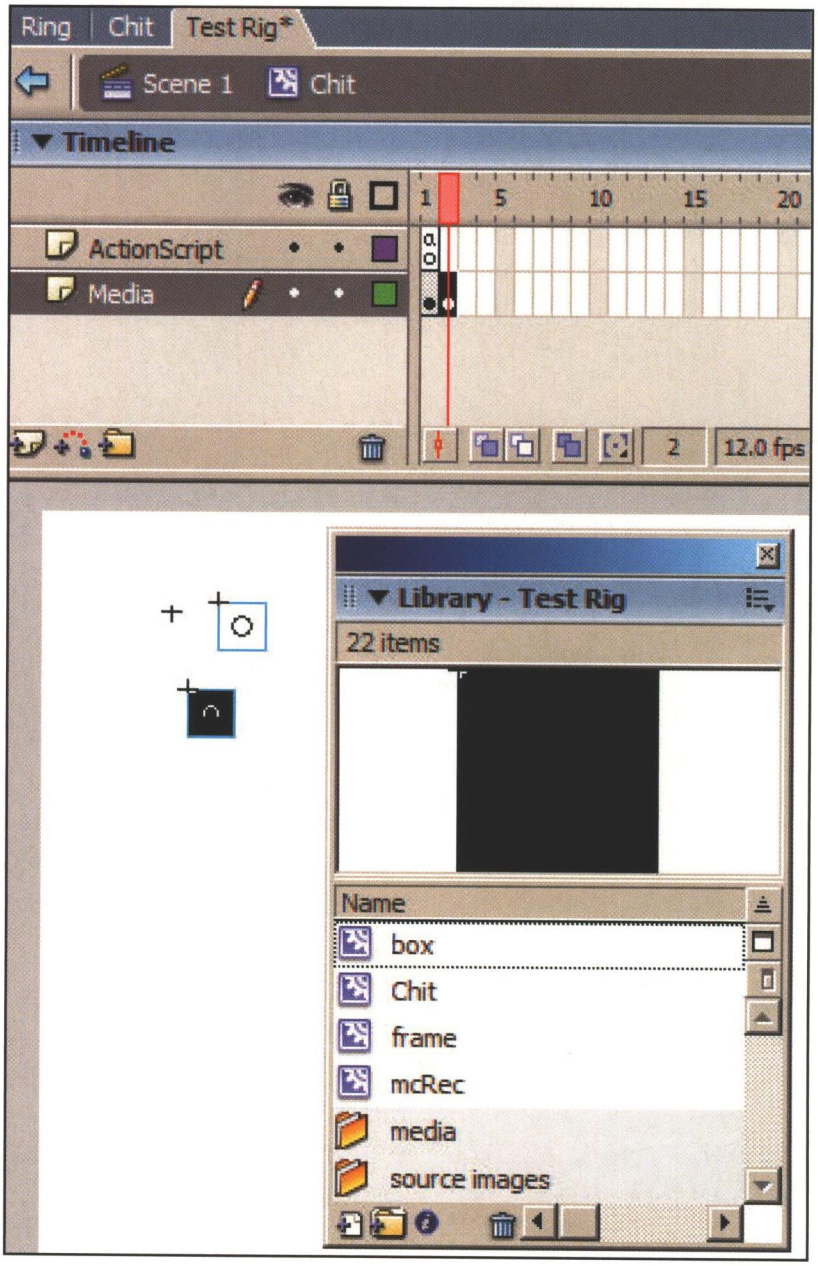
Part 1: The theory

Some important points before you get started...

Most components have a few basic requirements:

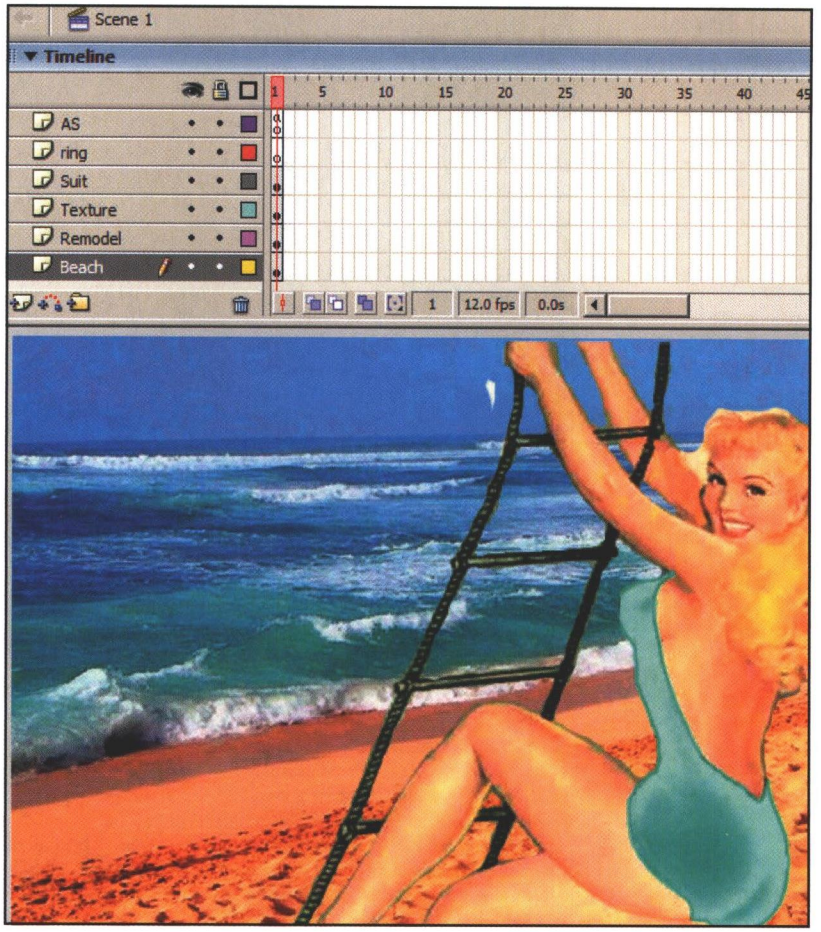
- A MovieClip with two layers and two frames.
- The first keyframe in the first layer is labelled ActionScript, and contains nothing more than a 'stop( );' script.
- The first keyframe in the second layer has a bounding rectangle to help *Flash* draw the component in the library. You don't need to do this if you're not attaching media with code. The second keyframe (see image) in the second layer has all of the media you'll be dynamically attaching.
- The MovieClip name should start with a capital letter. The linkage and class names should be set to match the MovieClip. Class names do not end in '.as'.
- MovieClips do not become components until you apply the component definition (right-click on the bottom of the list).
- Components are not complete until you have either compiled them or converted them into 'SWC' format. We will be compiling components in this tutorial.
- All components have a standard template. You will not need to use all of the template for all of your components.

Note that there are lots of other tutorials that cover the template in more detail on the Macromedia Website. This is simply what you need to know to get started.



Part 2: Laying out the stage and component media

Begin by creating your symbols and getting everything ready for the code...

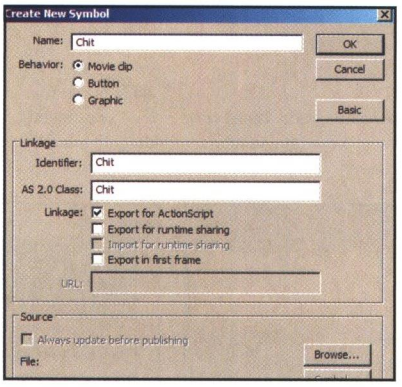


1 Create a movie and save it in a directory by itself. Create six layers. Get the beach and remodel from the completed example's library on the cover CD and drag them onto the layer with corresponding names. You need to build the ring component so that layer remains empty. Drag the texture map MovieClip onto its layer and position it over the remodel's bathing suit. Place the MovieClip suit in its layer and position it so that it covers the remodel's suit and the texture map. Give the suit the same instance name as its layer.

In the AS layer add the following script:

```
this.suit._alpha=50;
```

This sets the texture map to show through the suit to give the gradient effects.



2 Create a symbol in your library and call this Chit. Make sure it's capitalised in the same way as the class file Chit.as on the cover CD. In the Property dialog box, set the Linkage Identifier as Chit and the Class as Chit. (Do not use the '.as' in the class name.) Deselect 'Export in first frame.'

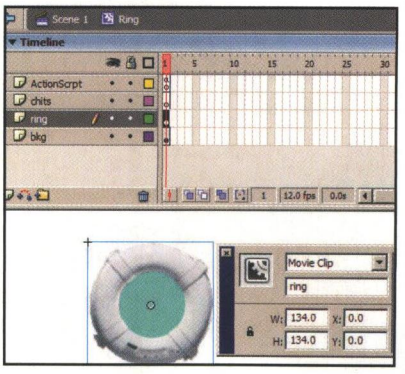
As described in Part 1, create two layers in Chit. Label the top layer as ActionScript and the bottom layer as Media. In the ActionScript keyframe place a stop();. In the Media layer place a keyframe in the second frame.

Open the library of the completed tutorial and drag the MovieClip's box and frame onto the second keyframe of the Media layer. The location doesn't matter. Open the properties and make sure that the linkage matches the MovieClip name in the library and that neither has a class attached.

Open a new MovieClip and name it Ring. You should set up the properties as described in the theory section (Ring for name, Ring for linkage and Ring for class). You can drag the RingMedia MovieClip out of the completed tutorial onto the stage. Delete it from the stage – you should find it in the library.

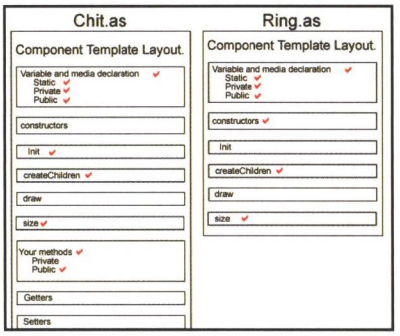
You should now have the stage and two potential components ready for code!

3 Set up the Ring MovieClip as illustrated below. Place a 'stop();' script in the AS layer. You need to create a backdrop for the ring so that the user doesn't click through to the stage. Use any graphic shape you like, as long as it fills the centre of the ring.



Part 3: Writing the component code into the templates

Now your components are ready for adding all the necessary code...



1 Make sure that the Chit.as and the Ring.as are in the same directory as the *Flash* movie you laid out in Part 2. Open the templates (if you're not using *Flash MX 2004*, you'll need an external editor). We'll only be using the areas of the templates marked, and we'll begin our coding with the Chit.as file.

```
static var className:String = "Chit";
static var symbolName:String = "";
static var developer:String = "David Aaron";
static var objectives:String = "This is a colour chit to
  embed into another component";
static var publicMethods:String= "flip and flop";
static var version:String = "0.9.0.1";
private var frameColour:Color;
private var hFrameColour:Color;
private var boxColour:Color;
private var boundingRec:MovieClip;
private var box:MovieClip;
private var frame:MovieClip;
private var boxHex2Base10:Number;
private var frameHex2Base10:Number;
private var frameHighlightHex2Base10:Number;
private var frameStartColour:Number;
[Inspectable (defaultValue="0xFF0000") ]
public var pboxColour:String;
[Inspectable (defaultValue="0x999999") ]
public var pframeColour:String;
[Inspectable (defaultValue="0x333333") ]
public var pframeHighlightColour:String;
```

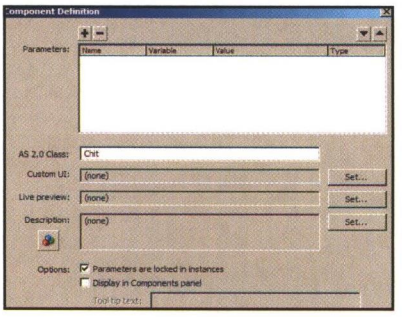
2 Each area of the code has a specific purpose. The execution order is init --> createChildren --> constructor (this is like code on your timeline) --> draw --> size. We declare our variables and media before our methods so that some values can be populated before we need them. This is what inspectable does for us in *Flash MX 2004 Professional* when it populates the Properties dialog box.

We usually do our variables in the order of static, private followed by public. Public variables can be read by anyone anywhere in the *Flash* context. Our variable declaration should look like this. This comes just after the class declaration 'class Ring extends mx.core.UIComponent {'.

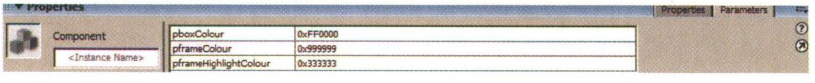
3 Static, private and public are colour coded (as shown above). When you enter the variables into the template, make sure the types match the part following the ':' shown here. You can customise the default colours in the inspectables to your own colour preferences.

4 You're now ready to turn the MovieClip into a component. Right-click on the MovieClip symbol Chit in the library and select Component Definition... When the Component Definition dialog box appears, type in the class name Chit and hit Enter. No other changes are necessary.

5 You can now go to Scene 1 and drag on a copy of Chit from the library (notice the new icon!). Look at the Properties dialog box (seen along the bottom of this page) and you should see the three variables we marked as inspectable in our class. The remainder of the code can now be added to the Chit.as class file.




```
function createChildren(Void):Void {
  //Set the events
  this.box.onRollOver = function() {
    frameColour = new Color(_parent.frame);
    frameColour.setRGB(_parent.pframeHighlightHex2Base10);
    delete frameColour;
  };
  this.box.onRollOut = function() {
    hFrameColour = new Color(_parent.frame);
    hFrameColour.setRGB(_parent.frameHex2Base10);
    delete hFrameColour;
  };
  this.box.onRelease = function() {
    //Set _parent._parent.Master colour
    _parent._parent.masterColour = _parent.pboxColour;
  };
}
```



```
function init(Void):Void {
    //Hide boundingRec and make sure it cannot receive any events
    boundingRec._visible = false;
    boundingRec._width = 0;
    boundingRec._height = 0;
    // Attach box and frame --> Do not worry about the location
    attachMovie("box", "box", 1)
    // See Note 4 about the depth settings
    attachMovie("frame", "frame", 2);
    //Note 5 Hex and based ten numbers
    boxHex2Base10 = Number(pboxColour);
    frameHex2Base10 = Number(pframeColour);
    pframeHighlightHex2Base10 = Number(pframeHighlightColour);
    //Set default colours
    setColour(boxHex2Base10, box);
    setColour(frameHex2Base10, frame);
    //Set box offset
    box._x += 2;
    box._y += 2;
    //frame is attached at 0,0 by default
}
```

```
//Private
private function setColour(arg1:Number, arg2:MovieClip):Void {
    boxColour = new Color(arg2);
    boxColour.setRGB(arg1);
    delete boxColour;
}
//Public
public function flip(Void):Void {
    setColour(pframeHighlightHex2Base10, frame);
}
public function flop(Void):Void {
    setColour(frameHex2Base10, frame);
}
```

```
class Ring extends mx.core.UIComponent {
  static var className:String = "Ring";
  static var symbolName:String = "";
  // You can use this._name="A Name" in init for a fix instance name
  static var developer:String = "David Aaron";
  static var objectives:String = "This is a container component to hold other components";
  static var version:String = "0.9.0.1";
  //Private
  private var changeColor:Color;
  private var currentColor:Number;
  private var temp:String;
  // Public String
  [Inspectable (defaultValue="suit") ]
  public var targetClip:String;
  [Inspectable (defaultValue="0xFF3399") ]
  public var masterColour:String;

  //Constructors
  function Ring() {
    temp = targetClip;
    this.onEnterFrame = function() {
      changeColor = new Color(_root[temp]);
      currentColor = Number(masterColour);
      changeColor.setRGB(currentColor);
      delete changeColor;
    }
  }
  //Required by Component architecture
  function init(Void):Void {
    if (masterColour == undefined) {
      masterColour = "0x00E8E4";
    }
  }
  function createChildren(Void):Void {
  }
  function draw(Void):Void {
  }
  function size(Void):Void {
    super.invalidate();
  }
  //Application based methods
  //Private
  //Public
  //Getters
  //Setters
}
```

6 The yellow block hides our boundingRec (see above image). The green and aqua blocks attach our MovieClips that were in frame 2 to frame 1 so they can be seen. The purple box converts the data that was inspectable into a number format. (While ActionScript doesn't care if our numbers are base 10 or hexadecimal, the Properties dialog box does. Any numerical data that is not in base 10 format needs to be brought in as a string and converted into a number. Think of it as a 'Feature.') We call a private method setColour twice. The method requires two arguments, a MovieClip and a number representing a colour. We will write setColour later – this applies the colours from the property box to our squares. The blue box at the bottom offsets our square by two pixels so that it appears in the middle of our frame MovieClip. MovieClips are attached at X=0 & Y=0 by default so our frame MovieClip does not need an X or Y position.

7 The createChildren method (see image on the left) is where all interactivity is controlled. The three events – onRollOver, onRollOut and onRelease – are all associated with the MovieClip box. The coding method should look very familiar to anyone who has written *Flash MX* code. In the onRelease event there is a special reference to the _parent. _parent.MasterColour. This variable was not declared with the other variables because it will be part of the Ring component.

Experienced *Flash MX 2004* coders generally do not code events like this when writing professional components. The current best practice recommended by Macromedia is to use listeners to monitor change events through the mouse or keyboard. If you run 'trace variables' under the debug menu when you're testing your code, you'll notice that *Flash* adds these to your code.

```
Movie Clip: Target="_level0.instance1"
Variable _level0.instance1.
addEventListener = [function
 addEventListener ].
```

If you're very comfortable with ActionScript, you can make the changes, otherwise no one will know because once a component is compiled the code is hidden from spying eyes!

8 The custom methods section (see top-right image) follows the createChildren method. The private method has been called already in the init method and is used to change the colour of MovieClips. The two public methods, flip and flop, are not required, but would enable the visible rollover states to be triggered by code outside of the component. It's always a good idea to think forward when developing a component since they are often reused with surprising frequency in new contexts.

Test your component. If it picks up your colour preferences from the Properties panel and the frame MovieClip changes colours when you move the mouse over it, you're ready to compile your clip. Right-click on your component and select 'Convert to Compiled Clip'.

9 Open the MovieClip Ring. It's time to bring Chit and Ring together. On the chits layer drag out the compiled chit components five times and arrange them tastefully – the component should now be called Chit SWF. Give the Chit components instance names of your choice. Give each one a unique hexadecimal colour in the pboxColour in the parameters panel. You can change the frames from your default if you like. The Ring code is simpler than the Chit code so it has been added for you.

The constructor code (see above) runs as though it were entering the frame with the MovieClip. While there are other ways that are more code efficient, this approach is reliable and easier to script. The constructor polls the Chit components and checks to see if any have changed and then changes the MovieClip colour on the stage. Test the component and compile it!

10 You're now ready to drag the Ring SWF onto the stage. There's nothing left for you to do now but test it. ca-p

SUBSCRIPTION ORDER FORM

☐ **I would like to subscribe to *Computer Arts Projects* [13 issues] and receive...**

The Hidden Power of Photoshop CS book FREE!

BEST BUY ☐ **Direct Debit UK** **£50** (£12.50 every 3 months)

☐ **UK** **£54**

☐ **Europe [inc. Eire & N. America]** **£73.53**

☐ **Rest of World** **£86.53**

Order Code COSP61

Your subscription will start with the next available issue.

Your details

Title Initials

Surname

Address

Postcode Country

Tel no. (inc. code) (home/work)

Email address

Mobile no. (home/work)

We will use the contact details supplied to contact you about your subscription. Indicate here if you're happy for *Computer Arts Projects*, Future Publishing and its sister companies to contact you with other information/offers by: ☐ email ☐ mobile. If you're also happy for us to pass your details on to carefully selected companies, indicate here: ☐ email ☐ mobile.

Payment details

I understand that I will receive 13 issues over the next 12 months.

☐ Direct debit (£12.50 every 3 months)

☐ Mastercard ☐ Visa ☐ American Express ☐ Switch

Card no. ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐

Expiry date ☐☐☐☐ Issue number ☐☐

Signature

Date

Instruction to your Bank or Building Society to pay Direct Debits.

Originator's Identification Number 7 6 8 1 9 5

Please send this form to address shown

Media with passion
Future publishing ltd.
Cary Court,Somerton,
Somerset, TA11 6BR

DIRECT Debit

1. Name and full postal address of your Bank or Building Society branch

To: The Manager

Bank/Building Society

Address Postcode

2. Name(s) of account holder(s)

3. Branch sort code (from the top right hand corner of your cheque) ☐☐☐☐☐☐

4. Bank or Building Society account number ☐☐☐☐☐☐☐☐

5. Instruction to your Bank or Building Society

Please pay Future Publishing Direct Debits from the account detailed on this Instruction subject to the safeguards assured by the Direct Debit Guarantee. I understand that this instruction may remain with Future Publishing and if so, details will be passed electronically to my bank or building society.

Signature(s) Date

Ref No (Office use only)

Banks and Building Societies may not accept Direct Debit Instructions for some types of account.

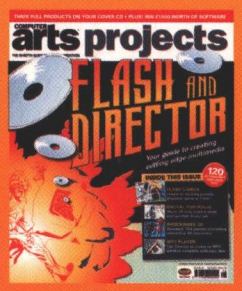
You may photocopy this blank form and then complete it.
You cannot fax or email direct debit forms.

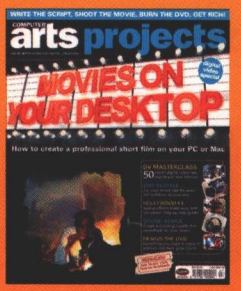
Return address

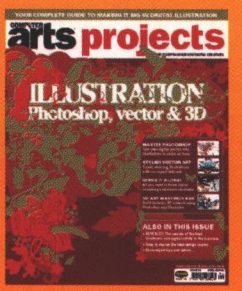
Please cut out your completed form along the dotted line and return it to: *Computer Arts* subscriptions, Future Publishing, Cary Court, Somerton, Somerset TA11 6TB, UK

Please indicate if you do NOT wish to receive relevant information about special offers or products/services from *Computer Arts Projects*, Future Publishing and its sister companies by ☐ post ☐ telephone. Please indicate here if you do NOT wish us to pass on your details to carefully selected companies to enable them to contact you about their products/services by ☐ post ☐ telephone.

OFFER ENDS 31 AUGUST 2004







Take advantage of our Direct Debit 'Best Price' and subscribe to *Computer Arts Projects* for as little as £12.50 every three months.

As well as spreading the cost of your subscription, you'll save £28* off the shop price and get a copy of *The Hidden Power of Photoshop CS*, worth £27.99, absolutely FREE!

* Saving is based on 13 issues bought on the UK newsstand and Direct Debit Best Price.

Stock is limited, so call our subscriptions hotline now or order online at **www.computerarts.co.uk**. Please quote **COSP61** when you contact us.

ALTERNATIVELY YOU CAN FILL IN THIS FORM AND POST IT BACK TODAY

COMPUTER arts projects

FREE GIFT!

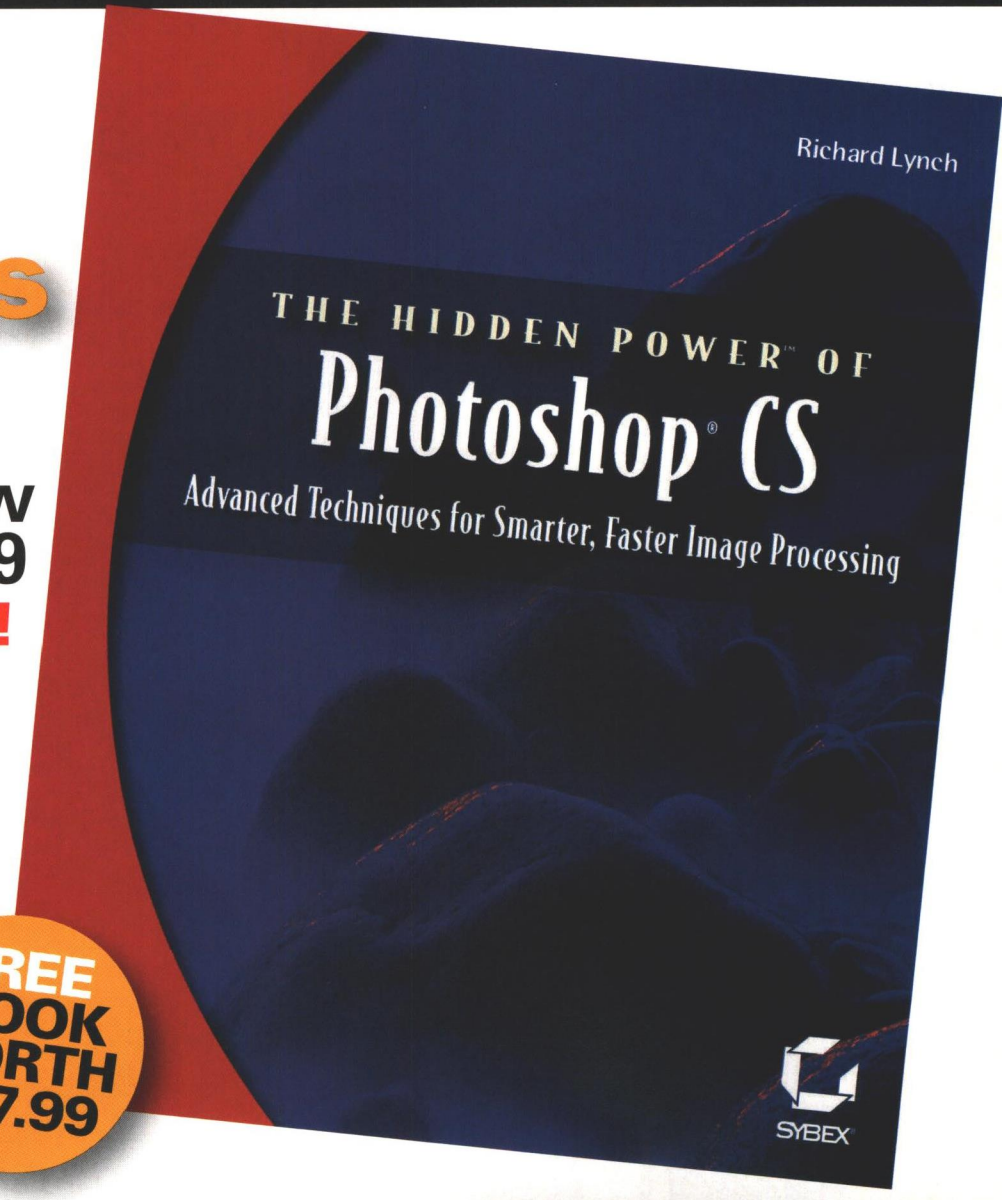
SUBSCRIBE TO COMPUTER ARTS PROJECTS AND RECEIVE THE HIDDEN POWER OF PHOTOSHOP CS* **ABSOLUTELY FREE!**

Subscribe to...

COMPUTER arts projects

...and get this new book worth £27.99 ABSOLUTELY FREE!

If you are an experienced user or a professional who wants to take advantage of the real power that lies inside *Photoshop*, this is the book that shows you how to tap the source.



FREE BOOK WORTH £27.99

CALL OUR SUBSCRIPTIONS HOTLINE NOW AND QUOTE ORDER CODE COSP61 TO TAKE ADVANTAGE OF THIS GREAT OFFER

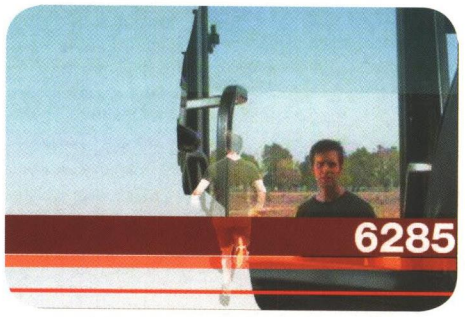
CALL 0870 444 8455

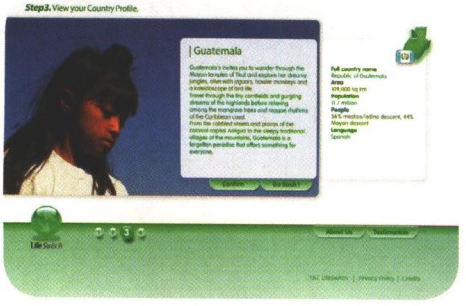
* Gift stock is limited. In the event of the gift not being available, Future Publishing reserves the right to offer a substitute gift of similar value.

OVERSEAS +44 870 444 8455

Hi-ReS!

"You give people the chance to communicate freely and things start to become interesting." Hi-ReS! lifts the lid on what makes interaction work...





Since 1999, Hi-ReS! has been blazing a trail through the world of interactive media, turning heads with a combination of dark humour, visual flourish and a willingness to explore what it means to be interactive.

"Hi-ReS! stands for handsome information, radical entertainment systems," explains Florian Schmitt, one half of the founding duo. "But sometimes we like to call it sponsored weirdness."

This volatile mixture has lured clients such as Lexus, HBO and MTV to the Hi-ReS! experience and led to the creation of groundbreaking sites soulbath.com, thepunisher.com and thethirdplace.com.

The challenge

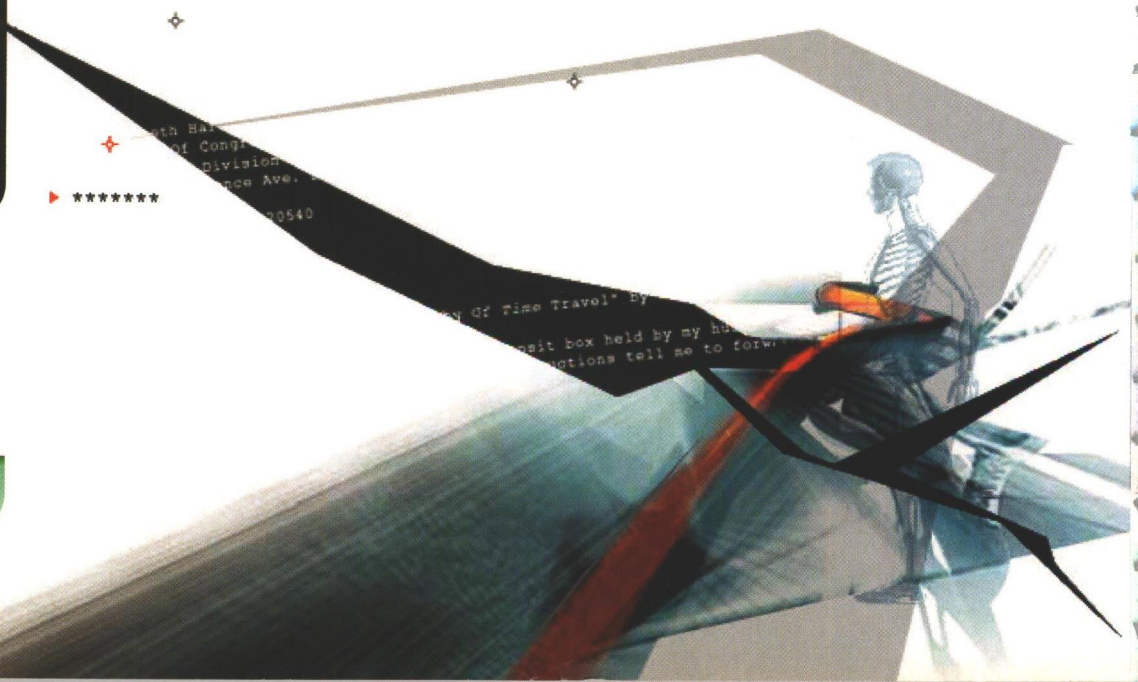
The rapid switch between playful and thoughtful that characterises its work also helps Hi-ReS! maintain an avowedly challenging edge — "It's our motto," explains Florian. "We want to challenge people."

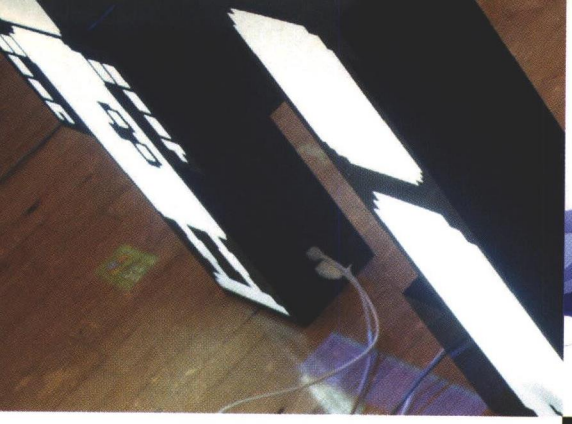
Florian, who judged the interactive category at this year's D&AD Awards, has coined his own phrase for what the company is doing: "I call it assisted discovery. It's about creating understanding, changing something through your interactions."

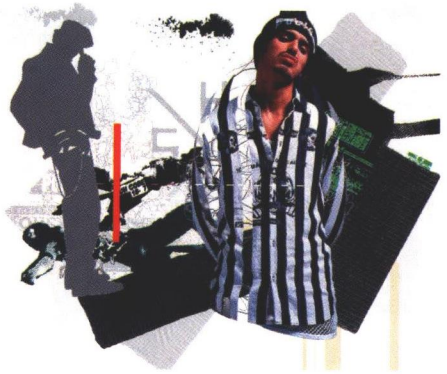
While he's clearly having a bit of fun, this does give some insight into what makes Hi-ReS! special, why its work is more than equal to the task of promoting films such as *Requiem for a Dream, Donnie Darko* and *The Punisher.* "It's about giving people the ability to influence the system," says Florian, as opposed to making the entire system obvious and therefore boring.

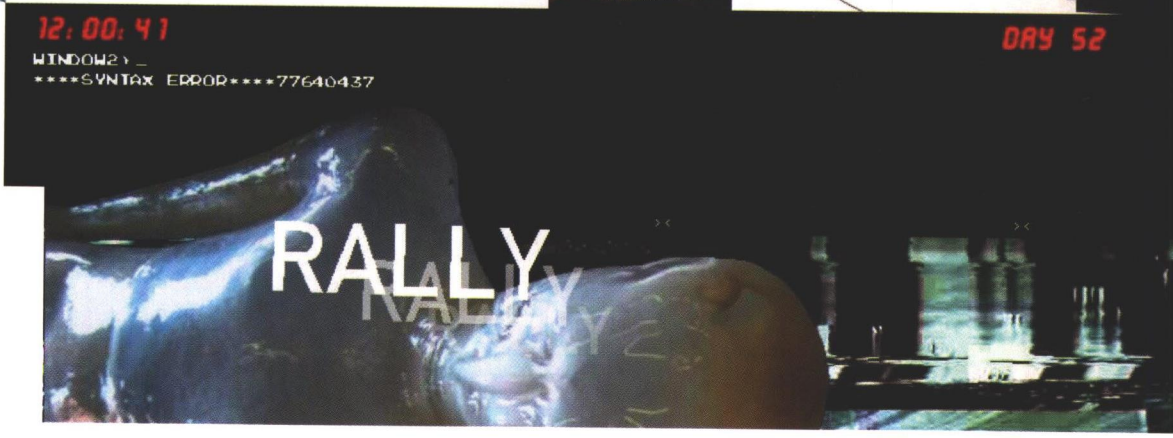
When they set up Hi-ReS!, Florian Schmitt and Alexandra Jugovic had one major ambition: "I think for us, the foremost goal was always to see work as play," says Florian. Unfortunately, there wasn't a barrage of eager punters looking for challenging multimedia content, because by then the Internet bubble was already starting to take on a mythical aspect.

So Alexandra and Florian decided to become their own first client, commissioning themselves to produce soulbath.com. "The idea to become our own clients was born out of the fact that no one was really waiting to give jobs to two Germans who had just arrived in London and hadn't done any interactive work." The way









Florian tells it you almost feel sorry for him, but the next chapter of the story fixes that.

The pair sent themselves a brief and, with a deadline set by the rapidly approaching expiry date on their trial versions of *Flash*, they set to work. "It was more of a fun project to keep us busy," muses Florian.

Celebrating malfunction

Soulbath is lo-fi meets hi-tech in a head-on crash, Florian explains by saying: "We wanted to do something that celebrated malfunction, that's what we were interested in at the time." Along with its associated exhibition of subversive banner ads, ClickHere!, the site won Hi-ReS! some well-deserved attention from, among others, *The New York Times*.

Hi-ReS! consistently produces sites that feel properly interactive, like they extend beyond your feeble clicking. "It's always difficult to know when to stop," complains Florian. "There's nobody here telling us to go home so we just keep going."

"For us, communication is the creation of understanding (or misunderstanding) through interaction between two entities." In developing a way to relate a story, Hi-ReS! is "always trying to anticipate this interaction in order to shape our narratives and thus the flow of the Websites." The outcome for each user is hard to predict, but that's how it should be.

Hi-ReS! continues to explore the nature of interaction with a combination of installation work and technological examination. Florian isn't entirely joking when he says, "We're researching 'alternative interactions' with 'the machine' at the moment."

At the Hi-ReS! studio in Shoreditch, work is currently under way on an installation and Website which will support a Barbican exhibition celebrating40 years of graphic design. "We're working on a gesture-based system at the moment," says Florian, a direction heavily influenced by the fact that, "It's amazing, if you give people the opportunity to draw a penis, they will!"

But Florian's happy with this. It's examining how people choose to communicate, taking their work in unexpected directions. "It's just one of those things. Because it's interactive, you can't predict what will happen every step along the way." ca-p

OPPOSITE PAGE, FROM TOP TO BOTTOM:
Hi-ReS! delivered titles, set, sound and logo design for MTV: New. The concept used was 'twisted wonders', described as "somewhere between fascination and being unsure."

Following the MTV:New project, Hi-ReS! is soon to unveil its work on an international style guide for MTV. Produced in tandem with fellow Shoreditch studio Precursor, it's intended to give disparate MTV studios a clear idea of the station's flavour.

The Wake forms one part of the Website Hi-ReS! produced for HBO's *Six Feet Under* series. This online experience is billed as a metaphysical pitstop, perhaps taking the Hi-ReS! 'sponsored weirdness' philosophy to its logical conclusion.

"The great thing about The Punisher is that he's a super hero, but he doesn't have any super powers – he's just really pissed off," explains Florian.

This site, for the charity Christian Aid, struck gold at the prestigious Cannes Lion Advertising festival this year. The site's glossy self-help parody is so good it has a genuine effect.

"We can do light and happy," insists Florian Schmitt. "It's just that the dark comes more naturally." The site that Hi-ReS! built for Richard Kelly's brilliant debut film, *Donnie Darko*, is proof that the second half of this comment is truer.

THIS PAGE:
Top left: With the Shiny Vistas installation projects, shown at this year's FEED Festival, scripts and interactions, sound and light are triggered by the viewer's presence.

Top right: The multi award-winning site for Sony's PlayStation concept saw Hi-ReS! reprise a curatorial role first seen on soulbath.com, collecting together the impressions of designers and artists from around the world to help define the concept.

Above left: "We're in the enviable situation of being able to turn down work we don't feel suits us," says Florian. Hi-ReS! works with clients, such as Diesel Style Lab, who are willing to give its 'challenging' philosophy a fair chance of success.

Above right: Pioneers in their own right, Massive Attack called on Hi-ReS! to produce these clips from footage of Nick Knight's cover shoot for *100th Window*. "It's about giving people the ability to influence the system."

Left: Hi-ReS! succeeds in making interactive experiences worthwhile because it intentionally avoids being obvious about what's on offer. LifeSwitch is a Website for Christian Aid.

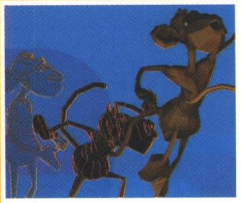
You can find out more about Hi-ReS! by visiting **[w]** www.hi-res.net.

CONTENTS

SECTION TWO: CUTTING-EDGE DIRECTOR

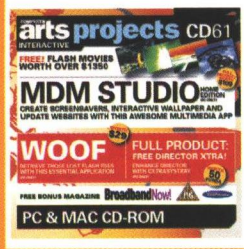
Director is the tool of choice for any professional involved in multimedia content creation. With our in-depth tutorials and expert tips, you'll be creating cutting-edge multimedia for CD-ROM, DVD or the Web in no time!

Director is a more powerful tool than you might think! Not only can you create stunning content for CD-ROM, but you can build dynamic and interactive multimedia for distribution on DVD and the Web. Over the following pages you'll learn how to control a 3D character using Lingo, *Director*'s native programming language, link DVD content to Web pages and create an MP3 player complete with play-lists. There's also 60 Lingo tips for your reference – whether you're a beginner or expert *Director* user...



p58 _ **TUTORIAL: Shockwave 3D**
Master the art of controlling 3D characters in *Director*. This in-depth tutorial tells you all you need to know about adding key controls, collision detection and cameras!

p66 _ **TUTORIAL: DVD in Director MX 2004**
In *Director MX 2004*, you can link DVD video events to Web pages. In this expert tutorial you'll learn how to add extra elements to DVD playback.


arts projects CD61
INTERACTIVE
FREE! FLASH MOVIES WORTH OVER $1350
MDM STUDIO
CREATE SCREENSAVERS, INTERACTIVE WALLPAPER AND UPDATE WEBSITES WITH THIS AWESOME MULTIMEDIA APP
WOOF
FULL PRODUCT: FREE DIRECTOR XTRA!
FREE BONUS MAGAZINE BroadbandNow!
PC & MAC CD-ROM

p96 _ **REGULAR: On the CD**
This month's cover CD-ROM is positively busting with fantastic multimedia resources. We've got three full products and over $1350-worth of *Flash* stock movies for you!

p74 _

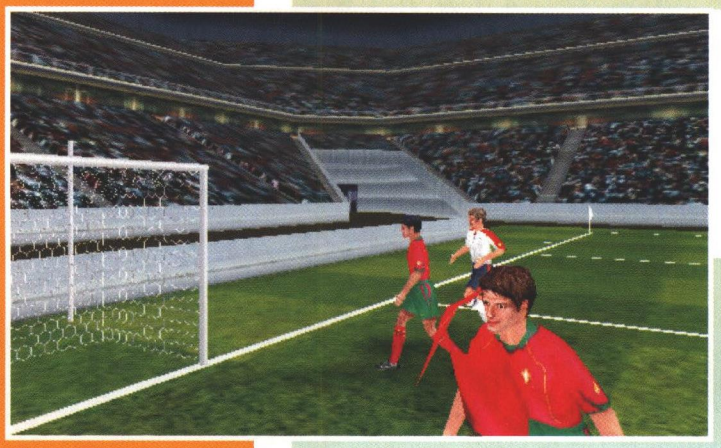
FEATURE: 60 Lingo tips

Director's native programming language can be tricky to master, but with our help, you'll soon get to grips with it. These 60 tips are for beginner, intermediate and expert *Director* users...

p84_

TUTORIAL: Create an MP3 player

Using *Director* and a bit of Lingo trickery, you can create a fantastic MP3 player complete with play-lists. And guess what? Here's how to do it!



p90 _

FEATURE: Making Virtual Replay

With its Shockwave 3D-based Virtual Replay service, the BBC has shown us the future of Web-based sports broadcasts. Here's how it was done...

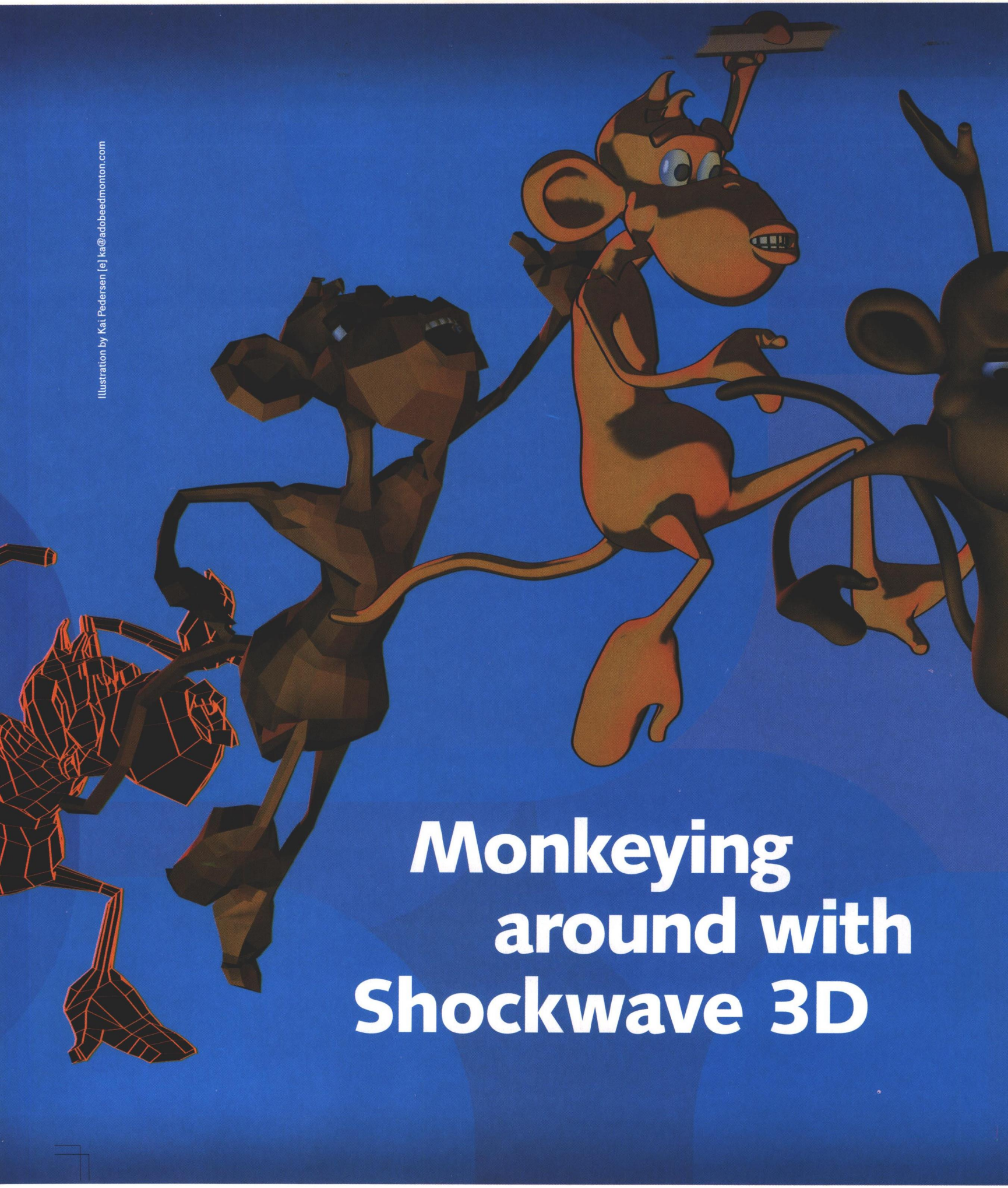


Illustration by Kai Pedersen [e] ka@adobeedmonton.com

Monkeying around with Shockwave 3D

Lights, camera, action and keyboard! Learn how to move a character around an environment in *Director*...

This tutorial will show you how to bring in a 3D scene, add lights and capture interactivity through the keyboard, then add collision detection. The project we're doing is very light on the maths, but we'll work in the X, Y, and Z axis. The scripts we're working on should work in all versions of *Director* from 8 onwards. If you're already coding in the new JavaScript model, you should be able to port these scripts without any major changes to the code between the 'on' and 'end' handlers. We're going to import a 3D scene, layout our event handlers, add our code, and then test it.

This guide assumes that you've written simple behaviours and know about the cast, channels, sprites, and where to find the Shockwave 3D window. Ours is a very simple Shockwave 3D scene, but you can apply the techniques here to more complicated models.

Thanks to Dean Utian and William Yeung of the Faculty of the Built Environment, University of New South Wales, Sydney, Australia. See **[w]** www.fbe.unsw.edu.au/learning/director/ or **[e]** d.utian@unsw.edu.au. Additional expertise from David Aaron at **[w]** david.aaron@thatplanet.net.

You'll find all the files needed to complete this tutorial in the Software\Tutorials\Tutorial files\Tutorial files\All\Shockwave 3D folder on the cover CD.

Part 1 : Controlling 3D characters

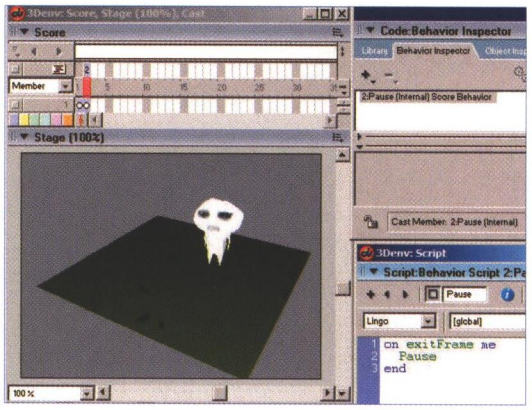
Get started by using basic code to control your 3D character...

1. Setting up your workspace

Set up *Director* and prepare your movie for the 3D character...

Open *Director MX 2004* and make sure you can see the stage, casts, score, and Behaviour palette. You may want to close any windows or palettes you don't need. In the score, double-click in the second script frame (frame 2) and add a pause script.

Director will provide you with the ExitFrame event. Just type Pause in-between the handlers (experienced Lingo developers know that they can write the behaviour first and drag it onto the frame script or use the Behaviour palette). Drag the cast member Scene into the first channel. You can shorten the frames in the channel to frame 2 if you want to be tidy.

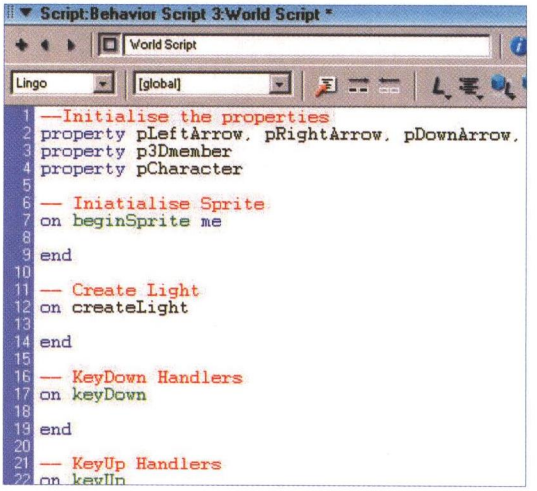


2. Lay out your behaviour

Lay out all the handlers in Lingo before you add any further code...

Click on an empty cast member and create a new behaviour. Call this World Script. You need to lay out all of the handlers that are required before you add any code.

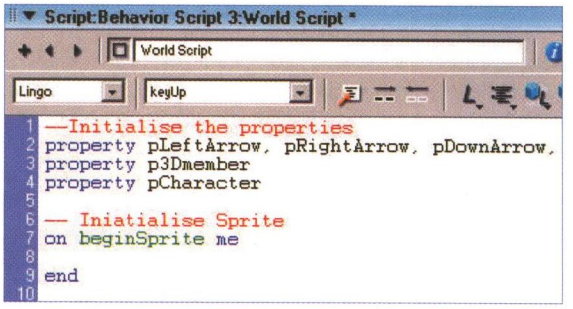
The properties at the top of the script are all of the necessary properties. The 'on beginSprite me' sets up our world, and the names of the other handlers should be very clear.



3. Setting up the interactivity

Here, we set up the key states so your character moves when you press certain keys...

You need to initially set up your key states. The properties are not actually the key states; rather they are flags defined in the initial property declaration. Set all of them as FALSE on the keyUp event and test for particular keys in the keyDown event. (Note: In the past, some keyboards in some locations have not matched these keypressed numbers. If you're going to work on a non-QWERTY keyboard, it's important to test this on the localised settings.)



4. Setting up the character movement

Next, using the code below, set up your character to move using the characterMove method...

You also need to set up the movement for the character and the frame exit script. So you need to set up the 'on exitFrame' and create your call to the characterMove method. Then set up the 'on characterMove' where you process the actual changes to the position of the character. The pLeftArrow rotates the 3D character anti-clockwise along the Z axis, and the pRightArrow sets the rotation in a clockwise direction. The pUpArrow and pDownArrow only move the character along the Y axis. This is because the X axis is handled by the rotation along the Z axis (really!) and is effectively part of the character. You only need to send the little man forward or backwards and the rotation takes care of the X.

```
-- What Happens in a frame
on exitFrame
  characterMove
end

-- How to move the character around
```

6. Bringing it all together

Your character won't move yet – first you need to finish up the on beginSprite handler and add the behaviour...

If you test your behaviour now, you'll find it won't work. Why? Because, first, you haven't finished the 'on beginSprite' handler where all the scene elements are brought together. Second, you haven't dragged the behaviour onto the sprite. Let's finish the beginSprite handler first. You need to tell the sprite that p3Dmember is this sprite, and then reset the world. The reset is critical to this procedure otherwise the object will be deleted. You then need to set the pCharacter property to the model you want to move. Then call your light with createLight. The last thing to do is drag your behaviour onto your sprite in channel 1 and test it.

```
-- Iniatialise Sprite
on beginSprite me
  p3Dmember = sprite(me.spriteNum).membe
  p3Dmember.resetWorld()
  pCharacter = p3Dmember.model("characte
  createLight
end
```

5. Throw some light on the issue

Create a light in *Director* to illuminate your Shockwave 3D scene...

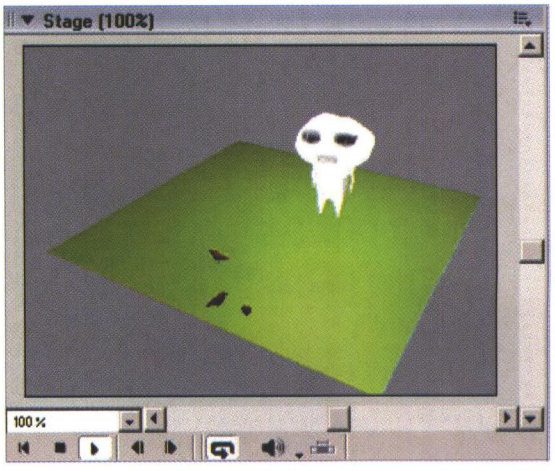
You're now going to finish the createLight handler. Attach the light to the scene, which is represented by the property p3Dmember. Put the 'Bulb Light' at 100 on the Z axis.

```
-- Create Light
on createLight
  p3Dmember.newLight("Bull
  p3Dmember.light("Bulb L:
  pCharacter.addChild(p3Dr
end
```

7. And finally...

You'll need to add a reset handler so the user can take the character back to its original state...

Let's add a reset handler to be triggered by the 'r' key. In your keyDown handler add 'if keypressed ("r") then reset Character'. You now need to add another method to the bottom of your script for the reset Character call.



Part 2 : Creating and controlling cameras

Using the following steps, add the Lingo code for controlling the camera...

1 Open the completed movie (3Denv_charMov.dir on the cover CD) and open the 'scene control' behaviour.

2 Add the following to the top of the script:

```
property pSprite -- reference to 3D sprite
```

3 Add the following to the end of the beginSprite handler: `setUpCamera`

4 Next, add the following code to the script (after the createLight handler):

```
on setUpCamera
-- create 2 new cameras
frontCam = p3Dmember.newCamera("frontCam")
characterCam = p3Dmember.newCamera("characteCam")
-- frontCam settings
-- set up the camera s vertical viewing angle (D8.5)
frontCam.projectionAngle = 50
```

The projectionAngle (or fieldOfView) is the vertical viewing angle of the camera, indicated by the angle formed by two rays: one drawn from the camera to the top of the projection plane, the other drawn from the camera to the bottom of the projection plane.

If you're using *Director MX* (or *Director MX 2004*), use fieldOfView instead of projectionAngle, which becomes obsolete. fieldOfView is also used for QTVR.

```
-- position the camera
frontCam.transform.position = vector(150, -300, 200)
```

The default position of any newly created camera is vector (0,0,250).

```
-- set rotation of camera
frontCam.transform.rotation = vector(60,0,30)
```

The default rotation of any newly created camera is vector (0,0,0).

```
-- characterCam settings
-- set position of the camera to position of the character
characterCam.transform.position = pCharacter.transform.position
-- set rotation of camera
characterCam.transform.rotation = vector(70,0,180)
```

Rotate the camera 180 degrees about the 'z' axis so it points from behind the character, and 70 degrees about the 'x' axis so it points slightly to the ground. A rotation of 90 degrees about the 'x' axis points the camera directly forward; greater than 90, it points up.

```
-- move camera up and back
characterCam.translate(0,30,300)
-- set the camera of the sprite to the frontCam pSprite.camera = p3Dmember.camera("frontCam")
-- -- make cameras children of the character model
pCharacter.addChild(frontCam, #preserveParent)
pCharacter.addChild(characterCam, #preserveParent)
```

By making the cameras children of the character model, the cameras will move with the model. So when you play the movie, the character will always stay in the middle of the scene and the environment will appear to move.

```
end
```

5 Now that you've created and set up the two cameras, create a script to switch between them. Add the following statement to the keyDown handler:

```
if keypressed("c") then changeCamera
```

6 Next, you guessed it, you need to create the changeCamera handler. Add the following to the script:

```
on changeCamera
-- check the sprites camera and switch
if pSprite.camera = p3Dmember.camera("characterCam") then
pSprite.camera = p3Dmember.camera("frontCam")
else
pSprite.camera = p3Dmember.camera("characterCam")
end if
end
```

7 Finally, test your movie.

Part 3 : Using the collision modifier

Here's where things get interesting – learn all about collision detection...

1 Open the movie 3Denv_cameras.dir and open the 'scene control' behaviour.

2 Add the following to the end of the beginSprite handler: `createWall`

3 Now you need to define what the createWall does. As the name suggests, you create a wall, which will be used for collision detection with the character. Before you write the script, you'll need to open the image brickmap.bmp from your cover CD, and import it into your movie. This will be used as a texture for the wall.

4 Add the following to the script:

```
on createWall
-- create a box model resource for the wall
wallRes = p3Dmember.newmodelresource("wallRes",#box)
  wallRes.height = 10
  wallRes.length = 50
  wallRes.width = 150

-- create a model from the model resource
wallMod = p3Dmember.newmodel("wallMod",wallRes)
-- locate the wall in our scene wallMod.translate(-50,-20,25)
-- create a texture for the wall, using the image brickMap
wallTexture = p3Dmember.newTexture("brick",#fromCastmember, \
member("brickMap"))
-- assign the texture to the shader for the wall wallMod.shader.texture = wallTexture
-- scale down the texture tile the bricks along the wall
wallMod.shader.textureTransform.scale  = vector(0.2, 0.4, 1)
```

The image has just been scaled to a width of one fifth (0.2) of the original and the height to two fifths (0.4). By default, the textureRepeat' property is set to TRUE so that tiling will occur. If the following statement is included, the texture will not tile:

```
wallModel.shader.textureRepeat = FALSE
```

There are other ways you can control texture mapping in *Director* including:

textureMode specifies how the texture is mapped onto the surface of the model

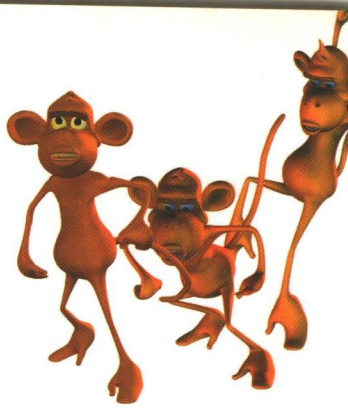
textureModeList property specifies textures for layers other than the first layer

```
end
```

5 You can now play the movie and move the character around – the character can move through the wall. Set up the collision detection using the collision modifier. Add the following to the end of the beginSprite handler: `createCollionDetect`

6 Add the following to the script:

```
on createCollisionDetect
-- add collision modifier to the "wall"
p3Dmember.model("wallMod").addmodifier(#collision)
-- set bounding geometry for collision detection to bounding box of model
p3Dmember.model("wallMod").collision.mode = #box
-- resolve collision for wall
p3Dmember.model("wallMod").collision.resolve = TRUE
-- add collision modifier to the character
pCharacter.addmodifier(#collision)
-- set bounding geometry for collision detection to bounding box of model
pCharacter.collision.mode = #box
-- resolve collision for the character
pCharacter.collision.resolve = TRUE
end
```



By attaching the collision modifier to objects in a 3D world, you can detect collisions between all the objects to which the modifier has been applied. Once the collision modifier has been attached to a model, the following collision modifier properties are accessible:

enabled (collision) enables you to turn off the collision detection (by setting the value to FALSE) while the modifier is applied. The default value is TRUE.

resolve indicates whether collisions models stop when they collide with each other. If the resolve is set to FALSE, then that model will continue to move. The default value is TRUE. In our example, we could have left out the resolve statements. Try changing the character resolve to FALSE and see what happens.

immovable indicates whether a model can be moved as a result of collisions. This default value is FALSE.

mode (collision) indicates the geometry used for collision detection and could be:

#mesh uses the actual mesh geometry of the model's resource. It is more precise but usually slower than #box or #sphere.

#box uses the bounding box of the model. It is useful for objects that can fit more tightly in a box than in a sphere, such as our wall. **#sphere** is the fastest mode, because it uses the bounding sphere of the model. It is the default value for this property.

7 Finally, test your movie. Note that the collision modifier is useful for relatively simple applications only. It can be very processor intensive, especially if there are complex models and a large number of modifiers attached to objects. The recommended technique for collision detection of more complex models is to use the modelsUnderRay command.

Part 4 : Using modelsUnderRay

Create a simple mesh around your character to aid in collision detection...

1 Open the movie 3Denv_collision1.dir from your CD and open the 'scene control' behaviour.

2 Add the following to the top of the script:

```
property pCharBoundingBox -- reference to character s bounding box
```

Often when setting up collision detection between complex models, a simplified mesh is created around the model to define the collision area. This is what we're going to do with our character model.

3 Add the following to the end of the beginSprite handler: `createCharacterBoundingBox`

4 Add the following to the script:

```
on createCharacterBoundingBox
-- create modelResource
charModRes = p3Dmember.newModelResource("charModRes",#box)
  charModRes.height = 25
  charModRes.width = 25
  charModRes.length = 50
-- create model and reference to it
pCharBoundingBox = p3Dmember.newmodel("CharBoundingBox",charModRes)
pCharBoundingBox.worldposition = pCharacter.worldposition
pCharacter.addchild(pCharBoundingBox, #preserveworld)
boxShader = p3Dmember.newshader("transparentShad",#standard)
boxShader.transparent = TRUE
-- set opacity of shader 0 is invisible
boxShader.blend = 50
pCharBoundingBox.shaderlist = p3Dmember.shader("transparentShad")
end
```

In the above script, a bounding box was created for the character. Its position was set to the same location as the character model. The box was made a child of the character so it would always move with it. A shader was created with the transparent property on. When the transparent property is on (TRUE), you can set the blend property to less than 100 per cent. Setting the blend to 0 will make the model invisible. For now, set it to 50 per cent so you can see the bounding box.

5 In the beginSprite handler, delete the line createCollisionDetect and add the following to the end of the handler.

```
-- register the member for regular timeMS events in order to respond to
-- user input and resolve camera collisions
-- i.e. after specified time segments
activate collisionDetect handler
p3dMember.registerForEvent(#timeMS,#collisionDetect,me,1000,10,0)
```

In the above statement, 'me' is the scriptObject parameter and indicates the collisionDetect handler is in the same script as the registerForEvent command. 1000 is the begin parameter and indicates that the first time the collisionDetect handler is to be activated will be one second (or 1000 milliseconds) after the registerForEvent command has occurred. 10 is the period parameter and indicates the subsequent time interval (in milliseconds) for the collisionDetect handler to be activated. 0 is the repetitions parameter and indicates the #timeMS event will occur indefinitely. Using 0 for repetitions makes the period parameter insignificant (it will be ignored).

6 Delete the createCollisionDetect handler and replace it with the following:

```
on collisionDetect me
-- create a list to store collision data created by modelsUnderRay
-- cast a ray to the left
collisionList = p3dMember.modelsUnderRay(pCharacter.worldPosition,\
-- pCharacter.transform.xAxis,#detailed)
```

In the above statement, a list is created (collisionList) to hold the information generated by the modelsUnderRay command. #detailed is used for the levelOfDetail parameter and will return a list of property lists, each representing an intersected model. #distance is one of the properties that will appear on the property list, which, in this case, represents the distance from the character to the point of intersection with the model.

```
-- if models picked up by ray, then activate custom handler
-- checkForCollision and send the collisionList as a parameter.
if (collisionList.count) then me.checkForCollision(collisionList[1])
-- cast a ray to the right
collisionList = p3dMember.modelsUnderRay(pCharacter.worldPosition,\
pCharacter.transform.xAxis,#detailed)
-- if models picked up by ray, then check for collision
if (collisionList.count) then me.checkForCollision(collisionList[1])
-- cast ray forward
collisionList = p3dMember.modelsUnderRay(pCharacter.worldPosition,\
-- pCharacter.transform.zAxis,#detailed)
-- if models picked up by ray, then check for collision
if (collisionList.count) then me.checkForCollision(collisionList[1])
-- cast ray backward
collisionList = p3dMember.modelsUnderRay(pCharacter.worldPosition,\
pCharacter.transform.zAxis,#detailed)
-- if models picked up by ray, then check for collisions
if (collisionList.count) then me.checkForCollision(collisionList[1])
end
on checkForCollision me, thisData
```

This handler is activated when a model is picked up by modelsUnderRay. The statement we used above:

```
me.checkForCollision(collisionList[1]) - - dot syntax
```

is equivalent to

```
checkForCollision me, collisionList[1] - - verbose syntax
```

So, collisionList[1] is assigned as a value to the parameter thisData.

Using modelsUnderRay continued...

```
-- grab the #distance value from the
collisionList dist = thisData.distance
-- check if distance is less than width of
bounding box if (dist < pCharBoundingBox.
resource.width/2) then
```

Width/2 has been used because the ray starts from the centre of the box (hence the distance to the edge is half the width).

```
-- get distance of penetration
diff = pCharBoundingBox.resource.width/2 -
dist
-- calculate vector perpendicular to the
wall s surface
-- to move the character (using the
#isectNormal property)
tVector = thisData.isectNormal * diff
-- move the character in order to resolve
the collision
pCharacter.translate(tVector,#world)
end if
end
```

7 Play the movie. If you want to remove the bounding box, change the blend to 0.

Part 5 : Moving a model over a contoured surface

Let's make your character move over complex terrain...

1 Open the terrain.dir movie from your CD, copy the 3D cast member, and paste it into your completed movie from the last tutorial (3Denv_collision2.dir). The 3D cast should be pasted over the original 3D cast member.

2 Open the 'scene control' behaviour and add the following code to the end of the characterMove handler:

```
-- move along terrain
-- create reference for terrain (ground)
  terrain = p3Dmember.model("Terrain")

-- store character s position
  charPos = pCharacter.worldPosition

-- cast a ray down
  collisData = p3Dmember.modelsUnderRay(char
Pos,vector(0,0,-1),#detailed)

  -- if model is picked up on ray
  if collisData.count then

    -- store total no of models detected by
the ray
    totalCount = collisData.count

    repeat with modelNo = 1 to totalCount

      -- check if over the terrain model
      if (collisData[modelNo].model =
terrain) then

        terrainPos = collisData[modelNo].
isectPosition

        -- find out the distance the
character should move up or down
        diff = (terrainPos.z - pCharacter.
worldPosition.z)+45

        -- move the character
        pCharacter.translate(0,0,diff,#world
)
      end if

    end repeat

  end if
```

In the above, the modelsUnderRay command was used to send a ray down and check the distance between the character and the ground. Since there could be other models on the ground, do a check through the list of models to make sure that you only move the character if the terrain model is on the collision list.

3 Play the movie and test what you've just created. You will notice that when you try to walk through the wall on the flat part of the ground, you are prevented from doing so. However, if you walk up the mound, you can walk through the wall. The reason for this is that the ray being cast forward is missing the wall. **ca-p**

Essential Shockwave 3D tips

There are a lot more features in Lingo 3D for you to play with. Here are a handful of hints that will make your life easier while you're learning...

1. Don't start with complex models imported from another program. Start by creating your own worlds in 3D Lingo. It will help you master *Director's* features.

2. Look for simpler solutions. Mix regular 2D media in with your 3D work. This improves performance and often makes your application more robust.

3. 3D worlds are object oriented. Parent -> Child relationships make the whole thing work. Before you start, make sure you have a clear understanding of which child belongs to which parent in your model.

4. In our example we clearly changed the X axis, but we did this through the rotation along the Z axis. We sometimes find it helpful to work with physical things on a checked tablecloth before we begin coding.

5. If you're a 2D developer, the final piece of advice we can offer is to be patient. Results do come, but they often emerge as your code develops.

THE INTERNATIONAL MAGAZINE FOR 3D ARTISTS

ISSUE 54 ON SALE NOW: PRICE £6.00

3D world
>>FILM>>TV>>GA
TIPS OPTIMISE YOUR PC FOR 3D
WIN! PURE render card WORTH £1,899
WIN!
PURE render card
WORTH
£1,899
SPECIAL 10-PAGE REPORT
SHREK 2
THE MAKING OF A BOX-OFFICE FAIRY TALE
PLUS The summer's hottest CGI movies! We round up the films to look out for and unearth the FX secrets of Troy
WORLD EXCLUSIVE FIRST TIME ON CD (PC ONLY)
GAMESPACE Light
Build your own models for videogames with this new learning edition of Caligari's popular 3D game-development package
FREE MANUAL Only available on this CD!
PLUS
EXCLUSIVE MAYA VIDEO TRAINING FROM 3D BUZZ
INCLUDED ON THE CD
RESOURCES WORTH OVER $1,000
THE DESIGNER'S TOOLBOX
3D EDITION: VOLUME ONE
• 5 models • 40 textures • 3ds max and Maya plug-ins
>> Interface requires Windows 98, Me, 2000, XP or Mac OS 8+ >> Some programs may require free registration over the internet >> Some software may not be available in all territories >> Program values quoted are prices when sold with original packaging
PLUS! BONUS MAGAZINE
Get issue one of BroadbandNow! free on this CD. Win one of ten broadband connections from fasthosts (UK only)
PG
CFJ09169
BroadbandNow!
10 DIRECTOR'S TRICKS
Proven ways to deliver quality animation on a tight budget
CREATE A VIRTUAL SET
FULL PROJECT Turn any photo into a 3D environment
AUGUST 2004
future
Media with passion
9 771470 438
next issue
next issue out 23/07/04

INSIDE THIS ISSUE

IN THE MAG Shrek 2 – monster 10-page report! Plus: we reveal the visual effects of *Troy*, and round-up the summer's hottest movies • **10 Director's Tricks** How to create quality 3D animation on a tight budget • **Create a virtual set** Make a full 3D environment using your own photos • **Optimise your PC for 3D** Read if you want to go faster! **ON THE CD: GAMESPACE LIGHT** Full product • The Designer's Toolbox 3D resources • *BroadbandNow!* issue one • *Rockfish* short

>> **TEN PAGES OF EXPERT TIPS**
3DS MAX • CINEMA 4D • LIGHTWAVE • MAYA • XSI

>> **SUBSCRIBE NOW!**
SAVE 42% • **+44 870 444 8451**

Combining DVD and Web pages

Using the new features in *Director MX 2004*, you can now insert DVD video directly into a Web page and then tie events from the DVD to the Web...

irector MX 2004 has some amazing new capabilities that enable you to combine DVD video content with Web pages. You can now make the Web page react to what is going on in the DVD, thanks to a nifty new *Director* Xtra called the DVD Event Manager.

There are a couple of important things to remember when working with this Xtra:

1) The user must have the DVD in their computer drive; if they don't, a big red X will appear where the video should be.

2) The Xtra doesn't differentiate between DVDs. If you create a presentation that triggers events at certain times of *Matrix Reloaded*, those same events will occur for any other DVD the user places in their drive.

Despite the minor drawbacks mentioned above, the new DVD Event Manager Xtra is an extremely powerful new feature that you'll enjoy using. If all this sounds a bit daunting, don't despair; it's actually quite easy and we'll walk you through the whole process.

Expertise provided by Ryan Carson. You can contact Ryan at
[e] ryan@carsonsystems.com or visit
[w] www.carsonsystems.com.

All the components needed to complete this tutorial can be found on the cover CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Director DVD folder.

Illustration by Kelly Boulton [w] www.other.com.au

Part 1 : The set-up

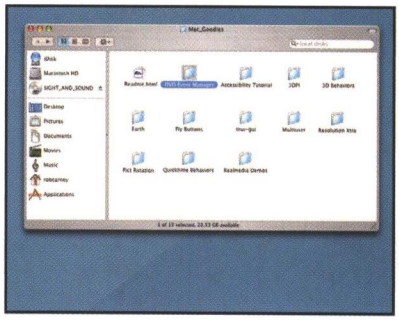
Boot up *Director MX 2004* and prepare your scene...

Be patient

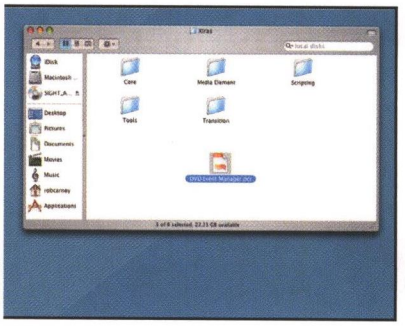
When using DVD and *Director*, don't get frustrated if it doesn't work straight away. Even we had to battle *Director* for a bit in order to get the DVD to play back. It's all about having patience!

X marks the spot

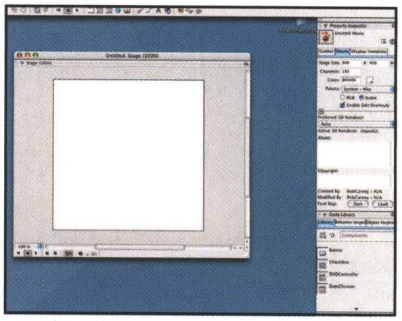
If you're having trouble getting the DVD to play back and are seeing a huge red X, make sure that there's nothing else accessing the DVD. For some reason, the Shockwave player doesn't cope well with this.



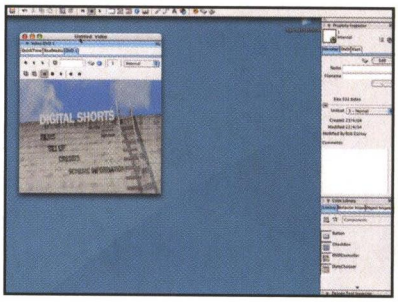
1 The DVD Event Manager isn't installed on the default install, so we'll need to do that now. Open the Goodies folder on your *Director MX 2004* CD. If you've lost the disc, you can download it from http://download.macromedia.com/pub/director/mx2004/goodies.zip.



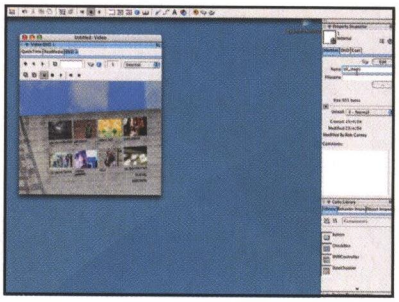
2 Drag the DVD Event Manager.dcr file from the Director CD into the *Director MX 2004* Xtras folder on your hard drive. You'll find it in the Configuration folder. The DVD Event Manager is now installed.



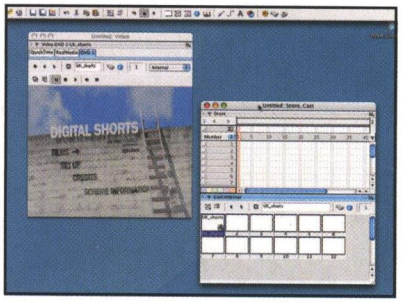
3 Open up *Director MX 2004*. The first thing you'll be doing is adding the DVD content as a cast member. Select Window>Property Inspector. Set the stage size to 400 by 400 pixels.



4 Next, select Insert>Media Element>DVD to open the DVD window. You can actually preview the DVD by clicking the Play button. In the next step, we'll give this cast member a name.



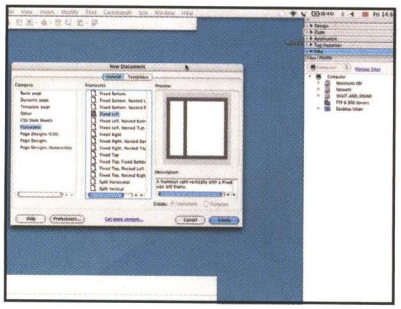
5 Open the Property Inspector by choosing Window>Property Inspector, and type in the name on the Member tab. We've chosen UK_shorts, because our DVD is a set of UK short films. The name really doesn't matter.



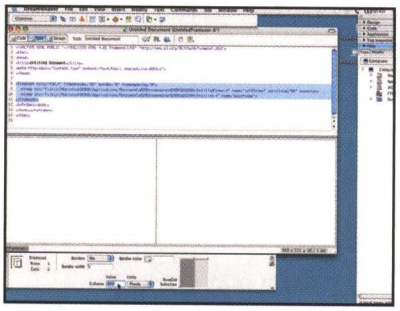
6 If you open the Cast window (Window>Cast), you can see the UK_shorts cast member. If you close the DVD preview window, you can easily re-open it by double-clicking on the UK_shorts cast member in the Cast window.

Part 2: HTML framework

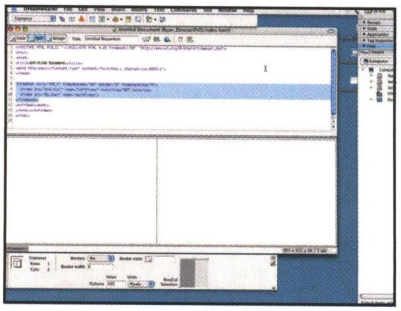
Create the HTML in *Dreamweaver* to work alongside your DVD footage...



1 Once you've set up the basic DVD cast member, you need to create the HTML framework for the movie to be housed in. To set up a simple HTML page with a left and right frame, open *Dreamweaver* and choose Framesets from the Category and Fixed Left from the Framesets options.



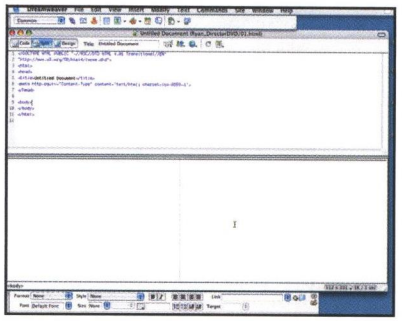
2 You'll see the HTML frameset open up in *Dreamweaver*. On the Properties Panel, type in 430 for the width (our DVD clip is 410 pixels wide). This will set the left frame at 430 pixels wide.



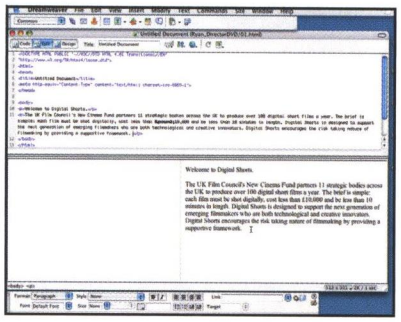
3 Save the HTML frameset as index.html (File>Save Frameset). Now you need to save the two HTML pages in the left and right frames. Choose File>Save All. Name the left HTML page "dvd.html" and the right page, "01.html".

QuickTime vs Shockwave

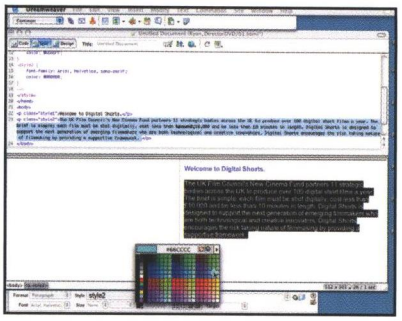
If you notice that your system is running very slowly, close down QuickTime. Shockwave and QuickTime seem to battle like dogs for system resources.



4 When you've finished creating the DVD *Director* file, which will be a Shockwave file, you need to place it in the dvd.html page. When the page opens, 01.html is going to be showing on the right. Let's modify that now. Click on the right frame to open 01.html.



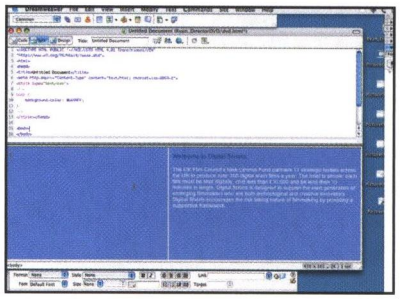
5 As our DVD is a series of UK short films, we're going to place a welcome message to the DVD. We've typed a welcome message explaining the DVD and its contents.



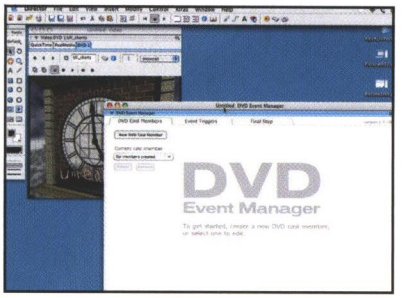
6 The page is a bit bland, so spruce it up by changing the colours of the background and text. Select the text and change the colour on the Properties panel (Window>Properties).

Part 3: Back into Director

Create more HTML pages then go back into *Director MX 2004* and add DVD event triggers...



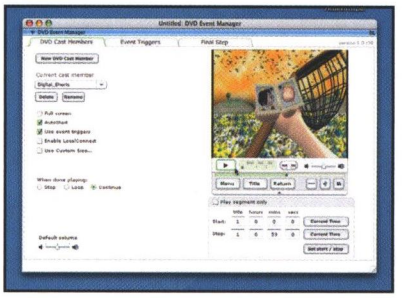
1 Create several other HTML pages to be shown when the film reaches certain places. Name these 02.html and 03.html. We do that later. For now, click on the left frame (dvd.html) and change the background colour (Modify>Page Properties).



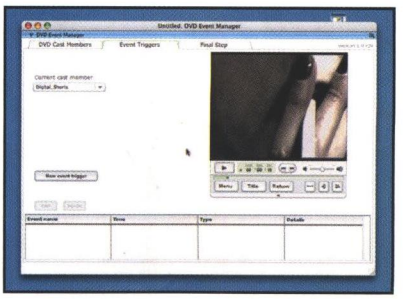
2 Now go back to *Director*. Select Xtras>DVD Event Manager from the menu. You'll notice that the window has three tabs: DVD Cast Member, Event Triggers, and Final Step.

3 Click the New DVD Cast Member button. Enter a logical name for the DVD Cast Member. We're going to use Digital_Shorts. Once you click OK, you'll see the DVD that is currently in the drive with play controls beneath it.

4 Don't be overwhelmed by all the options available to you on the left. They're actually quite straightforward. Make sure the Full Screen option is unchecked, because you're viewing the DVD inside a Web page. Go ahead and check AutoStart and 'Use event triggers'.



5 The right side of the DVD Event Manager interface enables you to control where the DVD starts and stops. This is a very powerful feature, because it enables you to create several different cast members of the same DVD, but with different sections being played.

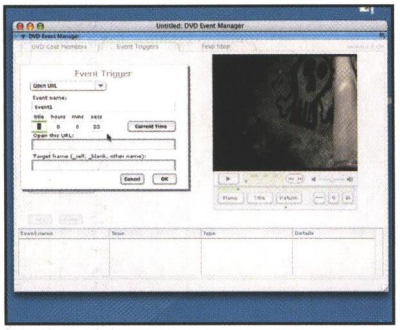


6 Now click on the Event Triggers tab. Even though this interface looks complex, its purpose is quite straightforward. You need to specify at what times during the DVD that HTML pages appear in the right frame.

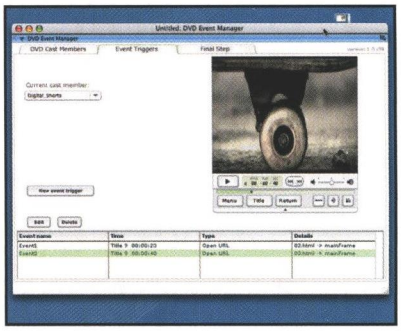
Stretch it

You can stretch the DVD Controller to match the size of your DVD content. Because it's made out of vectors it's fully stretchable.

7 Click on 'New event trigger'. This is used to set times in the DVD playback that will trigger events. Enter an Event name and enter 02.html in the 'Open this URL' text box. Enter mainFrame in the Target frame box.



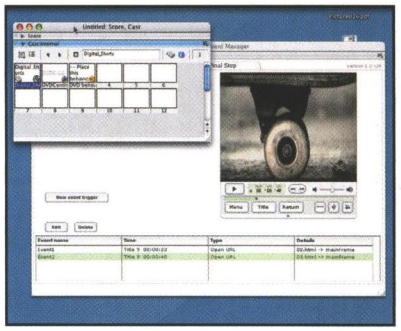
8 You need to specify the exact time that the event is triggered. You can hit play on the DVD controls and when it gets to the right spot, click the Current Time button. Alternatively, you can manually enter the title, hours, mins and secs in the text boxes. Now click OK.



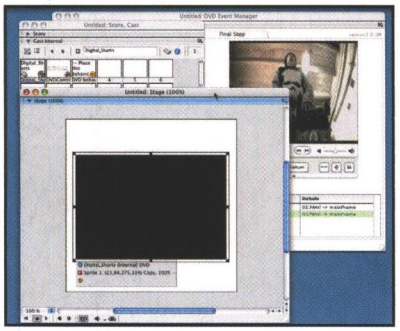
9 You can now see the Event1 event listed at the bottom of the page. You can select the event and edit it by clicking the Edit button or delete it by clicking the Delete button. Feel free to add as many events as you like by repeating steps 7 and 8.

Part 4: DVD controls

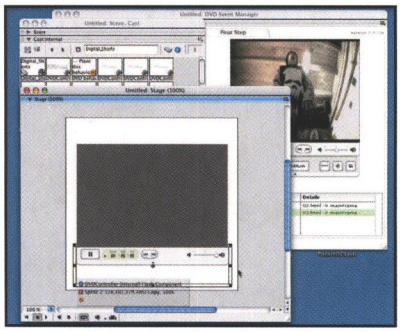
Bring your project together by adding a DVD controller and publishing your work...



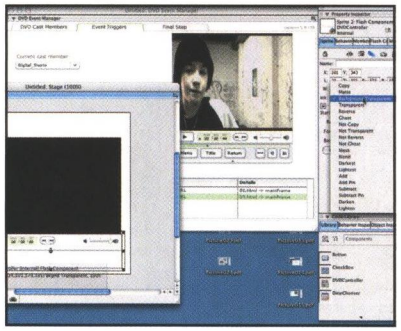
1 Now that you've added the event triggers, click on Final Step. You need to build the movie on the stage, before closing the DVD Event Manager window. Open the Director Cast by choosing Window>Cast.



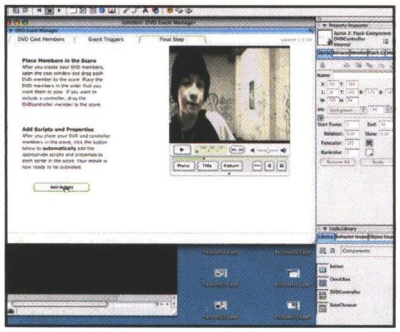
2 Drag the DVD cast member (ours is called Digital_Shorts) onto the stage. To open the stage, select Window>Stage from the menu. You'll see the DVD appear as a black box on the stage.



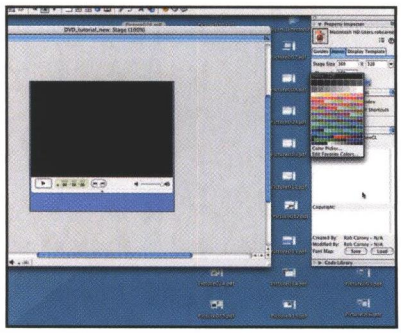
3 Now we need to place the DVD Controller onto the stage. This is a pre-built component that was created by the DVD Event Manager and placed in the cast. Open the Cast panel (Window> Cast). From the cast, drag the DVD Controller onto the stage and place it under the DVD.



4 The DVD Controller has a default background colour of white. If you'd like it to match the stage colour, select the controller on the stage and set its ink to Background Transparent in the Property Inspector (Window>Property Inspector).



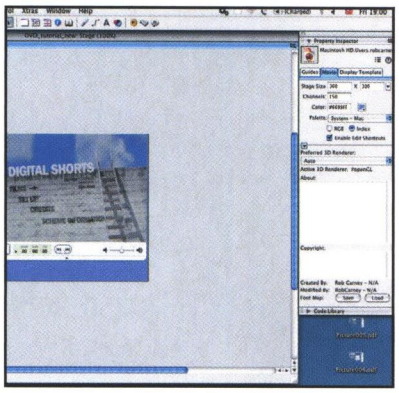
5 Having placed the DVD Cast Member on the stage and the DVD Controller underneath it, you can finish the DVD Event Manager. Return to the Final Step panel and click Add Scripts. When it finishes, you'll be told that the operation was successful. Now close the DVD Event Manager.



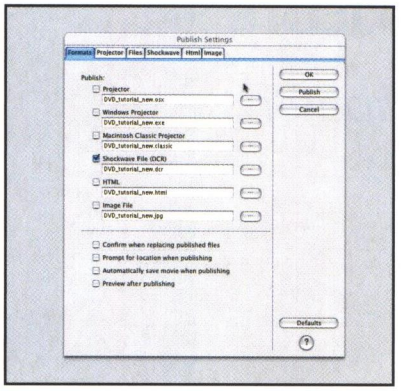
6 Now you need to set up the stage to look good. Open the Property Inspector (Window>Property Inspector) and click on the Movie tab. In the Colour text box, enter the Hex value for the background or use the colour picker.

Black screen

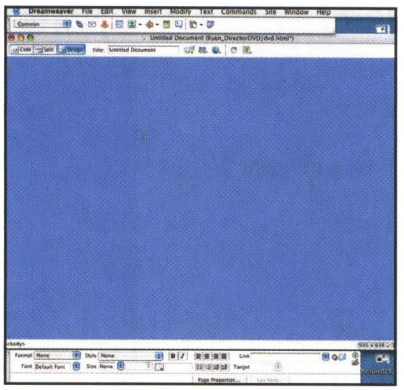
If you get a black screen where the DVD video should be playing, make sure that after you put the DVD Cast Member and DVD Controller on the stage, you went back to the DVD Event Manager and clicked Add Scripts.



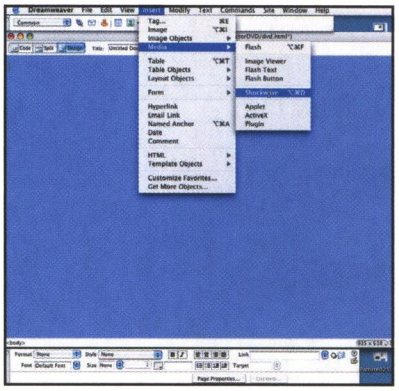
7 Next, enter the width and height of the stage. This is also on the Property Inspector panel on the Movie tab. Make sure you leave enough space for the DVD Controller (84 pixels high).



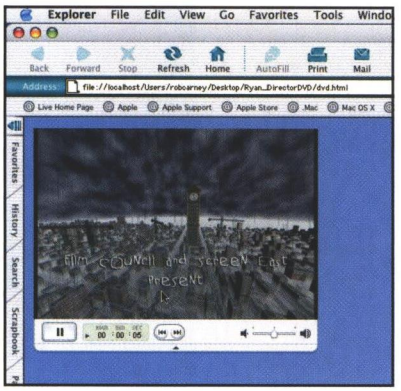
8 All you have to do now is publish the file into a Shockwave file. Choose Publish Settings from the File menu. Uncheck the Projector box and check the Shockwave File (DCR) box. The default name should be fine. Then click Publish.



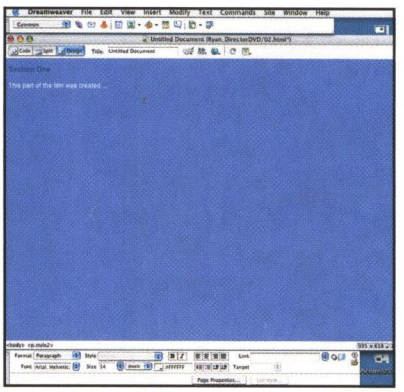
9 Quit *Director*. Move the .dcr (Shockwave) file into the same directory as your HTML files. Open up *Dreamweaver* again and open the dvd.html file. Make sure that the background colour is the same as the colour you chose for the stage in *Director*.



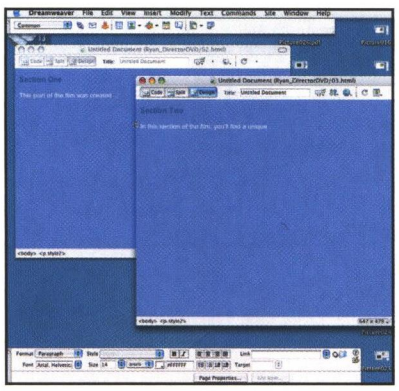
10 Choose Insert>Media>Shockwave from the menu and browse for your .dcr file. Now select the Shockwave icon and change the dimensions on the Properties panel. For our movie, it's 360 x 320.


Explorer File Edit View Go Favorites Tools Windo
Back Forward Stop Refresh Home AutoFill Print Mail
Address file://localhost/Users/robcarney/Desktop/Ryan_DirectorDVD/dvd.html
Live Home Page Apple Apple Support Apple Store .Mac Mac OS X
Favorites History Search Scrapbook

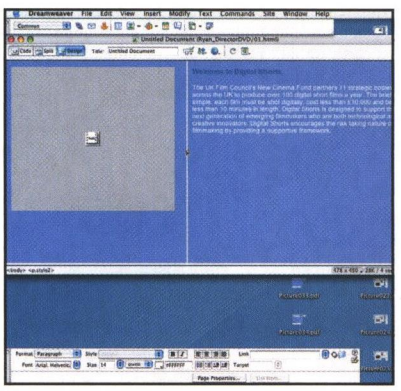
11 Go ahead and give the Shockwave file a test by hitting F12. This will open up Internet Explorer. If you get a big red X where the DVD should be, make sure you've closed *Director*.



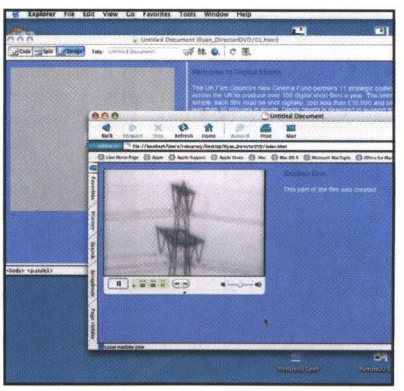
12 Now all you have to do is create the pages that you used in the DVD Event Controller trigger events. We used the names 02.html and 03.html. Create a new HTML document and save it as 02.html. Enter the text that you wanted to be displayed when the event was triggered.



13 We've entered text that explains what's going on behind the scenes of the film as the events are triggered. This is a great way to make viewing the DVD even more interesting.



14 Lastly, you need to tweak index.html slightly. Open it in *Dreamweaver*. Choose View> Visual Aids>Frame Borders. You can now re-adjust the left frame width to fit the Shockwave file.



15 And now you can give it a final test! With index.html open in *Dreamweaver*, hit F12 to view it in a browser. You'll see the right frame change to 02.html and 03.html as you specified in the DVD Event Controller. ca-p

SPECIAL NORTH AMERICAN OFFER

SUBSCRIBE TO COMPUTER ARTS PROJECTS

I want 13 issues of *Computer Arts Projects* for just $99

Your details

Title Initials

Surname

Address

Zipcode Country

Tel no. (inc. code) (home/work)

Email address

(home/work)

We will use the contact details supplied to contact you about your subscription. Indicate here if you're happy for *Computer Arts Projects*, Future Publishing and its sister companies to contact you with other information/offers by ☐ email ☐ mobile. If you're also happy for us to pass your details on to carefully selected companies, indicate here ☐ email ☐ mobile.

Payment details

Please fill in your credit card details. Note: payment will be debited in Sterling at the rate of exchange set at the start of this campaign in Jan 2004 (£54.61)

☐ Mastercard ☐ Visa ☐ American Express

Card no. ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐

Expiry date ☐☐☐☐

Signature

Date

Return address

Please cut out your completed form along the dotted line and return it to: *Computer Arts Projects* subscriptions, Future Publishing, Cary Court, Somerton, Somerset TA11 6TB, UK

ORDER CODE: COSUS61
OFFER ONLY APPLIES TO RESIDENTS OF USA OR CANADA

CLOSING DATE: 31 AUGUST 2004

Please indicate if you do NOT wish to receive relevant information about special offers or products/services from *Computer Arts Projects*, Future Publishing and its sister companies by ☐ post ☐ telephone. Please indicate here if you do NOT wish us to pass on your details to carefully selected companies to enable them to contact you about their products/services by ☐ post ☐ telephone.

arts projects
SUBSCRIBE TODAY
US AND CANADA ONLY
13 ISSUES FOR JUST $99
SUBSCRIBE TO COMPUTER ARTS PROJECTS AT OUR SPECIAL RATE OF $99 USD.
THAT'S UNDER $8 AN ISSUE!

Why subscribe?

- FREE DELIVERY TO YOUR DOOR
- NEVER MISS AN ISSUE
- RISK-FREE GUARANTEE

If you're not completely satisfied, you can cancel your subscription and receive a full refund on any unmailed issues

Other ways to subscribe

HOTLINE +44 870 444 8455 Quoting order code COSUS61
(lines open 24 hours a day)

ONLINE www.computerarts.co.uk/USAsubs

FAX +44 (0)1458 271 146 Complete and return the order form on this page

MAKE MUSIC NOW

THE WORLD'S BEST-SELLING MUSIC SOFTWARE MAGAZINE
PC X MAC
www.computermusic.co.uk
computer music
MAKE MUSIC
FREE! VOCODER PLUG-IN + 361 samples on the CD
THE ULTIMATE COMPUTER MUSIC STUDIO!
We show you how to build it without breaking the bank
TUTORIALS
Build an entire track from one sample!
Master the art of step sequencing
Spice up your drum loops
48 PAGES OF TUTORIALS
SOUND LIKE... PRINCE
He's back! We show you how to get his sound in software
REVIEWED
NI Guitar Rig ▸ TerraTec Axon ▸ Arturia Storm 3
Garritan Personal Orchestra ▸ Yellow Tools
Majestic ▸ Edirol PCR-1 ▸ AND MORE!
£5.99
CM76
AUGUST 04
ALL THE SOFTWARE YOU NEED TO MAKE MUSIC
computer music CD-ROM
MAC & PC
MAKE MUSIC NOW
CM76 AUGUST 04
EXCLUSIVE FULL SOFTWARE!
CM VOCODER
Go retro crazy with this amazing PC vocoder plug-in from the makers of FL Studio!
PLUS! The latest Mac and PC software demos
Spin Audio RoomVerb M2
Wave Arts FinalPlug
Zero-X BeatQuantizer
u-he Zebra 1.5
FREE! 361 ROYALTY-FREE SAMPLES
Koncept & Funktion Drum 'n' Bass Collection
Cyclick Step Sequencer Source Collection
FULL SOFTWARE
THE CM STUDIO
Everything you need to make great music now!

■ EXCLUSIVE SOFTWARE

Create crazy vocal effects with the awesome **CM Vocoder**. Plus, the pick of the latest demos and **361** royalty-free samples that you won't find anywhere else!

■ BUILD THE STUDIO OF YOUR DREAMS

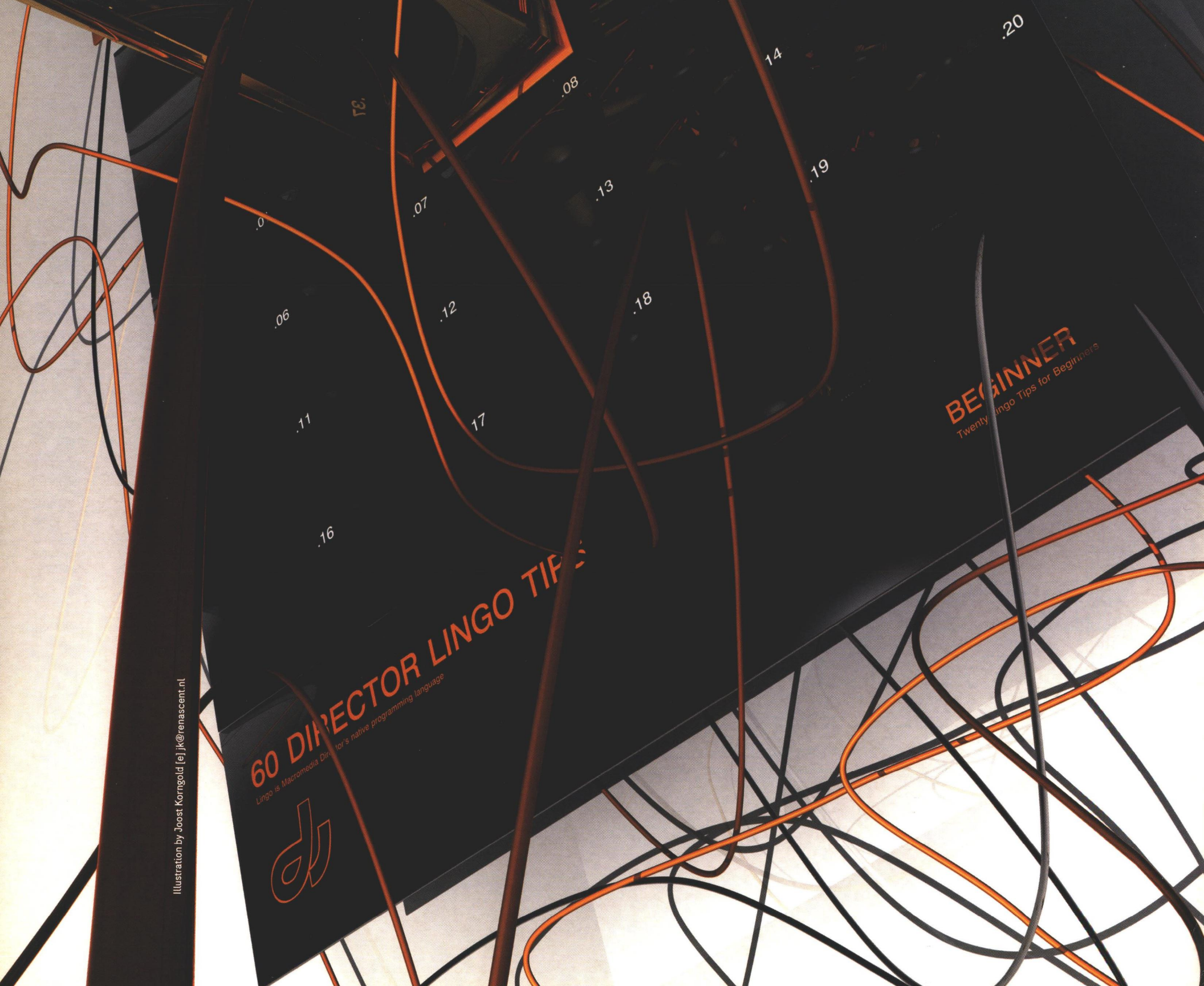
It doesn't cost much to turn your bedroom into a full-on recording facility

■ PLUS!

Sound like Prince ▸ Master your step sequencer ▸ NI Guitar Rig and Arturia Storm 3 reviewed ▸ Get creative with your drum loops

ON SALE JULY 20

Featuring a **FREE** virtual music studio on the CD every month, *Computer Music* is all you need to make incredible music on your PC and Mac


.08
14
.20
.07
.13
.19
.06
.12
.18
.11
17
.16
BEGINNER
Twenty Lingo Tips for Beginners
60 DIRECTOR LINGO TIPS
Lingo is Macromedia Director's native programming language
Illustration by Joost Korngold [e] jk@renascent.nl

60 LINGO TIPS

Lingo is the programming language for *Director*. Whether you're a novice or an expert, we'll show you some Lingo that'll make your life easier...

You can do things in *Director* using Lingo that you'd never have imagined possible. Our three categories of tips show you how. Beginner is for those just starting out with Lingo. You'll find tips on using buttons, making sprites move, using *Flash* components with Lingo and even a tip on controlling a DVD movie.

The Intermediate section is for people who've been using Lingo for a little while and are familiar with the basics. Did you know that you can use imaging Lingo to create a wave effect or that you could learn to stream your movies, control *Flash* sprites and even use JavaScript in *Director*?

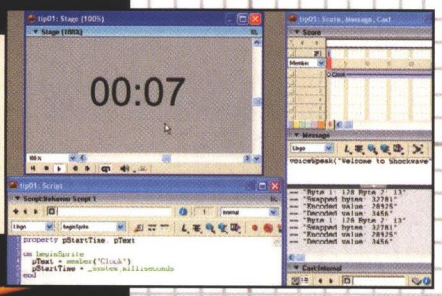
Finally, the Advanced section is for those who already know Lingo well but want to expand their knowledge by using 3D Xtra, the XML parser object, and the bitwise operators to encode your sensitive data. Even these tips are fairly easy to add to your code.

All the tips can also be found on the CD in the Software\Tutorials\Tutorial files\Tutorial files\All\Lingo tips folder – just open them in *Director* and copy the code into your own movies.

60 LINGO TIPS

TIP 1

Use some simple Lingo code to create an onscreen timer.

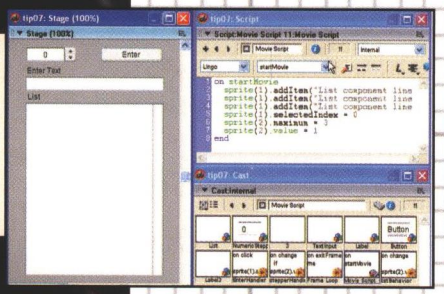


TIP 3

Use a property list to create a looping animated sprite.

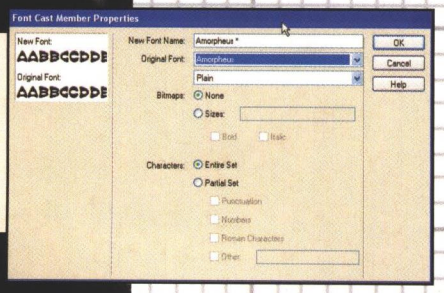
TIP 7

A TextInput component and a *Flash* button enables users to add to lists.



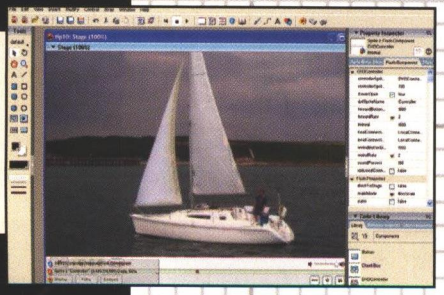
TIP 8

Embed special fonts if your movie is to be used on other computers.



TIP 10

Take complete control of a DVD by visiting the Macromedia Website.



1. Displaying a simple timer

You can display a timer by using the property _system.milliseconds. First, create a text member. Place this on the Stage and add a Behaviour by opening the Property Inspector and pressing the + button. Then create a beginSprite handler and an exitFrame handler. Initialise a property variable in the beginSprite handler and use this in the exitFrame handler to create a text version of the time since the movie started. Take a look at the file on the cover CD for more details.

2. Using a button

Use the Tool palette to add a *Flash* button to the Stage. Then add a text box, rename this 'message'. Add a Behaviour to the *Flash* button (click the button, click the Behaviour tab in the Property Inspector and press the '+' key).

```
property pCount
on click
  pCount = pCount + 1
  member("message").text =
"Clicked " & pCount & " times"
end
```

Add this code to the Behaviour. Every time you click the button, the message text will be updated.

3. An animated sprite

Here we use a property list to store the first and last frame of a looping animation. Then in the enterFrame handler for the sprite we set the appropriate frame and loop back to the first frame at the end:

```
  pAnim = [#id:me.spriteNum,
#cur: 1, #first:1, #last:12]

on enterFrame
  pAnim.cur = pAnim.cur + 1
  if pAnim.cur > pAnim.last then
pAnim.cur = pAnim.first
  sprite(pAnim.id).member =
member(pAnim.cur)
end
```

4. Moving a sprite using the keyboard

You can use the _key object to detect whether a key is pressed. The code that follows tests for the left arrow key. If this is pressed, set the flip direction for the sprite and update the horizontal location by reducing it by four:

```
  if _key.keypressed(123) then
    if pAnim.dir<>1 then
      sprite(pAnim.id).flipH =
true
      pAnim.dir = 1
    end if
    sprite(pAnim.id).locH =
sprite(pAnim.id).locH - 4
```

5. Using a Flash List component

We used our first *Flash* component in tip 2. In this example, we use a List component. To add a value to the list, give the sprite channel where it resides an addItem:

```
  sprite(1).addItem("List
component line 1")
```

6. Using a Flash Numeric Stepper component

Here, we add a Numeric Stepper component to the movie we had in tip 5. A Numeric Stepper has a minimum and maximum value and then the Up and Down arrows click between these values. You can add a Behaviour to the stepper as you would any sprite, and then add a handler for the change event, which happens when the displayed value changes.

7. Using a Flash TextInput component

By adding a TextInput component and a *Flash* button with a click Behaviour handler, we can add items that are entered by the user into the List component:

```
on click
  sprite(1).addItem(sprite(3).
text)
  sprite(2).maximum = sprite(2).
maximum + 1
end
```

Notice that we also increase the maximum value of the Stepper as we add a new item.

8. Embedding a font

If your movie is to be used on another computer and you've used special fonts, you should use embedded fonts. You can do this by using Insert>Media Element> Font... This opens a new dialog that enables you to select the font, the name for the

embedded font and whether to embed all or just a limited section of the font. The font is added to the Cast.

9. The Speech Xtra

To get Lingo to read a string of text is very easy, using the Speech Xtra. Just use:

```
voiceSpeak("The Speech Xtra is cool")
```

Anything you place in brackets will be spoken by the computer.

10. Controlling a DVD

You can control a DVD by following the instructions at http://www.macromedia.com/devnet/mx/director/articles/integrating_dvd.html. Or take a closer look at this example on the cover CD.

11. Starting and stopping QuickTime movies

Import a QuickTime movie using 'Link to External File' in the Media dropdown. Place this on the Stage. To control playback, set the movieRate of the QuickTime sprite to 0 to stop and 1 to play at full speed:

```
sprite(1).movieRate = 0
```

12. Rewinding and stepping through movies

You can display any frame of a QuickTime movie by using the movieTime property of the QuickTime sprite:

```
sprite(1).movieTime = 0
```

13. Film loops

When using animated sprites, film loops are useful. You can use them as a single frame while they display animated content. To create a film loop, highlight a set of frames and layers in the Score window. Select Insert>Film Loop. A dialog box appears asking you to name the film loop. *Director* stores all the Score data and cast member references as a new film loop cast member.

14. Display the date

You can use the _system method date() to get the current date:

```
_system.date()
```

15. Displaying the time

You can use the _system method time() to get the current time:

```
_system.time()
```

16. The pointToWord method

To find the word in a text sprite under a specific screen location, use the method pointToWord. This takes a point location as a parameter and returns the index of the word in the text sprite:

```
wordID = sprite(me.spriteNum).pointToWord(pt)
```

17. The pointInHyperlink method

A text member can contain hyperlinks by using the Text Inspector window. Just highlight some text and then enter the url that you want to link to. If you want to display a special cursor when the mouse is over a hyperlink then use the pointInHyperlink method.

```
sprite(me.spriteNum).pointInHyperlink(pt)
```

18. Colours using Lingo

Colours in Lingo are created using the colour method. This takes three parameters – a value between 0 and 255 for red, green and blue:

```
color(255, 0, 0) -- Bright red
```

19. Drawing lines with Lingo

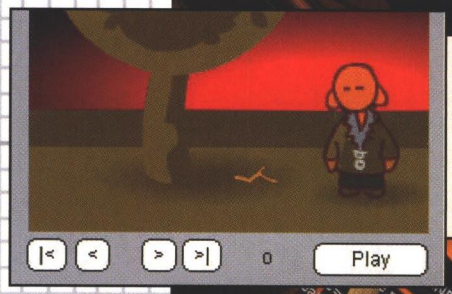
To draw with Lingo you use an Image object. Bitmap members, shapes and the Stage all have image properties that you can use as the Image object. To draw a line, you pass the start of the line, the end of the line and a colour:

```
imageObj.draw(x1, y1, x2, y2, color(0, 255,0))
```

20. Drawing shapes with Lingo

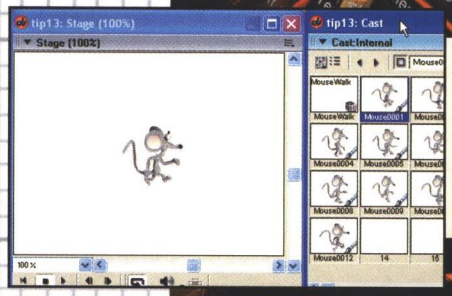
By replacing the last parameter with a list, you can use the same method used for drawing lines in the previous tip for drawing and filling shapes:

```
img.fill(rect(200, 100, 350, 300), \
    [#color:color(0, 255, 0), #shapeType:#roundRect, \
     #lineSize:2, #bgColor: color(0, 0, 0)])
img.draw(rect(200, 100, 350, 300), \
     [#color:color(0, 0, 255), #shapeType:#oval, #lineSize:2])
```



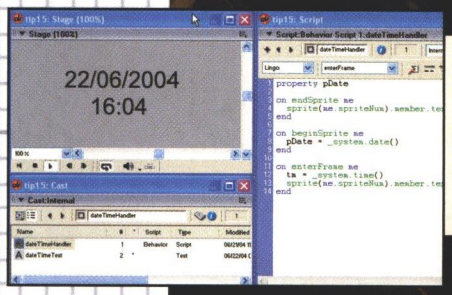
TIP 11

Place your movie on the Stage to set its start and stop functions.



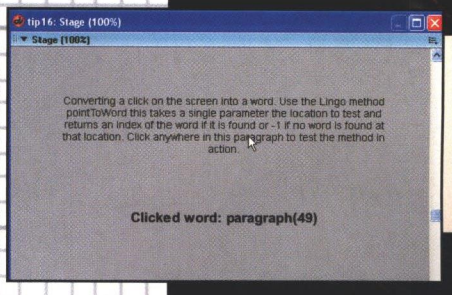
TIP 13

Film loops are useful for displaying animated content.



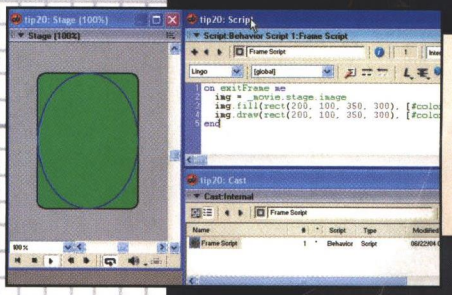
TIP 15

Displaying the date is a simple matter of using the _system method.


Converting a click on the screen into a word. Use the Lingo method pointToWord this takes a single parameter the location to test and returns an index of the word if it is found or -1 if no word is found at that location. Click anywhere in this paragraph to test the method in action.
Clicked word: paragraph(49)

TIP 16

Use the pointToWord method to find a word in a text sprite.



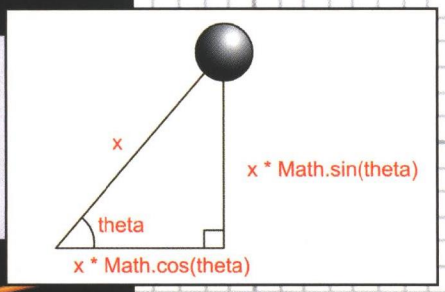
TIP 20

Drawing shapes is just an extension of the drawing lines method.

60 LINGO TIPS

TIP 1

You can calculate distances by using some simple calculations.

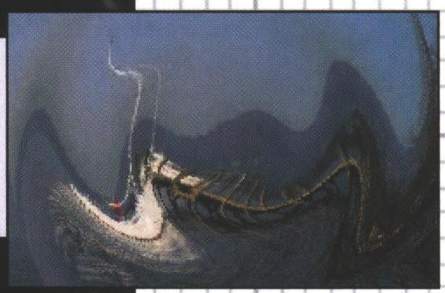


TIP 2

With imaging Lingo you can alter the pixels to create a wave effect.

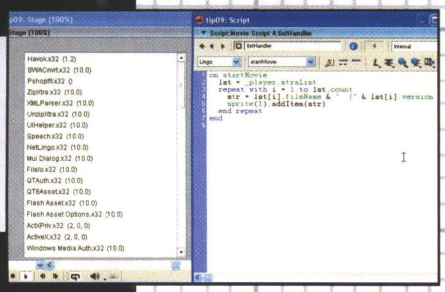
TIP 3

More dramatic effects can also be created with imaging Lingo.



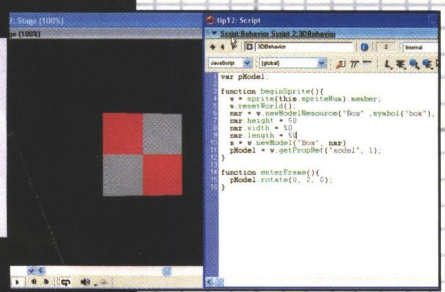
TIP 8

Find out exactly what Xtras are available using this code.



TIP 11

Access properties using Lingo and/or JavaScript.



1. Calculating distances

Calculate the distance between two sprites using the following Lingo:

```
dx = sprite(me.spriteNum).locH
- sprite(1).locH
dy = sprite(me.spriteNum).locV
- sprite(1).locV
dist = sqrt(dx * dx + dy * dy)
```

Use the square root of the sum of the square of the distance apart in the x and the square of the distance apart in the y.

2. Using imaging Lingo

Imaging Lingo such as copyPixels gives the developer access to the pixels in an image. Using this method, you can create some amazing effects. This one creates a wave running through an image:

```
imageObj.copyPixels(sourceIm
ageObj, destRect, srcRect)
```

3. Creating a spiral effect

In this example, we take the imaging Lingo one step further and create a spiral effect with the image. This sample is much slower because it works on each pixel in the image rather than using the image lines of the previous sample.

4. Displaying the current loading status

By passing the movie's url to the method getStreamStatus, you get a property list that enables the developer to display to the user the streaming status of the file:

```
url = _movie.path & _movie.name
statList = getStreamStatus(url)

if (statList.bytesSoFar > 0)
then
  percentDone =
integer((statList.bytesSoFar *
100.0001) / statList.bytesTotal)
```

5. Using mediaReady to control a movie

In addition to the total number of bytes downloaded, you can also use the mediaReady property of a *Flash* or *Shockwave 3D* cast member to determine whether to move the playback position:

```
if not member("conker").
mediaReady then
    _movie.go(_movie.frame)
  end if
```

6. Using the _key object

Use the _key object to read the keyboard:

`_key.keyCode` The numerical code for the last key pressed
`_key.key` ANSI value that is assigned to the key
`_key.controlDown` Returns TRUE if the Control key is being pressed
`_key.shiftDown` Returns TRUE if the Shift key is being pressed
`_key.commandDown` Returns TRUE if the Command key is being pressed
`_key.optionDown` Returns TRUE if the Option key is being pressed

7. Using the _system object

You can use the _system object to learn about the host computer:

`_system.colorDepth` Returns the bitdepth of the screen
`_system.deskTopRectList` Returns a list of rectangles
`_system.environmentPropList` Returns details about the operating system
`_system.milliseconds` Returns the milliseconds since switch on

8. Finding what Xtras are available

Use the _player object to learn what Xtras are available:

```
lst = _player.xtraList
repeat with i = 1 to lst.count
  str = lst[i].fileName & "  ("
& lst[i].version & ")"
  put str
end repeat
```

9. Using JavaScript

With *Director MX 2004* you can mix Lingo and JavaScript. Simply set the dropdown box in the Script window to JavaScript to start using the alternative syntax.

```
lst = _player.xtraList
for (var i=1; i<=lst.count;
i++){
  str = lst[i].fileName + "  ("
+ lst[i].version + ")"
  trace(str);
}
```

10. Using symbols with JavaScript

Lingo makes extensive use of symbols that are preceded with the hash symbol:

```
Lingo syntax:
state = #playing
```

In JavaScript you use the symbol method and pass the name of the symbol instead:

```
JavaScript syntax:
var state = symbol("Playing");
```

11. Accessing properties with JavaScript

If "News Items" is a text cast member then you access the data differently in JavaScript. In Lingo you would use:

```
member("News Items").
paragraph[1].word[2]
```

In JavaScript you use:

```
member("News Items").
getPropRef("paragraph",
1).getProp("word", 2);
```

12. Using JavaScript Arrays

The JavaScript equivalent of lists is the Array. You can create one using the new keyword:

```
myArray = new Array();
```

This has a number of useful methods, summarised in this tip. See the code on the cover CD for details.

13. Pointing one sprite at another

The maths operations used with JavaScript are more comprehensive than with Lingo. They are accessed via the Math object and include asin and acos, which are used in this useful tip for pointing one sprite at another.

14. Controlling a Flash sprite

Controlling a *Flash* sprite has never been easier with the new methods available in *Director MX 2004*. To control which frame number is displayed, simply use either:

```
sprite(n).gotoAndStop(frameNum)
sprite(n).gotoAndPlay(frameNum)
```

With the Stop version, the playback will stop at the frame; and with the Play version, play will commence at the frame.

15. Setting Flash variables

You can now access *Flash* variables and Objects directly. If your *Flash* sprite contains a variable called tree and a MovieClip called leavesMC that contains a variable called count, then these can be set using:

```
sprite(1).tree = "Oak"
sprite(1).leavesMC.count = 100
```

Note that if you set a *Flash* variable that doesn't exist in this way, it causes the Shockwave player to crash.

16. Calling Flash functions

Even functions inside *Flash* can be accessed and returned values used. If your *Flash* contains a function that adds two numbers, then you can use:

```
answer = sprite(1).add(4, 3)
put answer
```

The Message window would contain the answer *7*.

17. Flash calls Director

You can call functions in *Director* from *Flash* by using

```
getURL("lingo:hellofromflash");
```

in *Flash*, and then creating a handler in a Movie Script in *Director*:

```
on hellofromflash
   put "Flash says Hello"
end
```

18. Using 3D text

Creating 3D text is easy. Create a text member as usual then in the Property Inspector>Text tab set Display to 3D Mode. Now in the 3D Extruder tab set the properties you want. The text is displayed as a 3D world and can be manipulated as you would a 3D world. We'll bring you more on this later.

19. Sending a message to a sprite

Movie Script functions are available throughout your movie and can be called directly from a sprite behaviour. To call a sprite behaviour function from outside of the sprite, you use sendSprite:

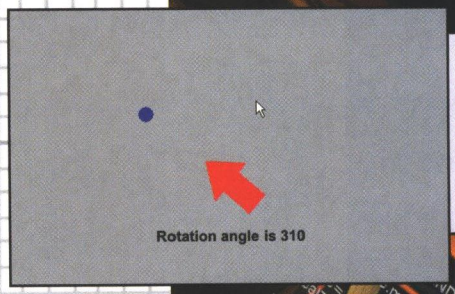
```
sendSprite(channel,
#functionName [, 1, 2, 3])
```

Parameters following the functionName are passed to the function.

20. Opening a message window in a browser

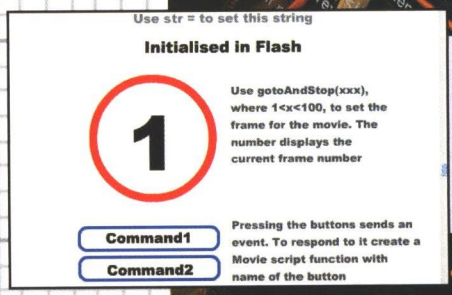
To test and debug a Shockwave movie inside a browser, it's sometimes useful to open the message window. This is easily done using:

```
  _player.debugPlaybackEnabled =
true
```



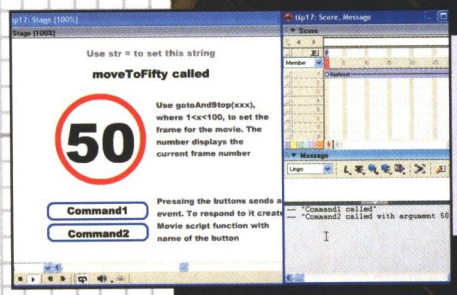
TIP 13

Make one sprite point at another using JavaScript commands.


Use str = to set this string
Initialised in Flash
1
Use gotoAndStop(xxx), where 1<x<100, to set the frame for the movie. The number displays the current frame number
Command1
Command2
Pressing the buttons sends an event. To respond to it create a Movie script function with name of the button

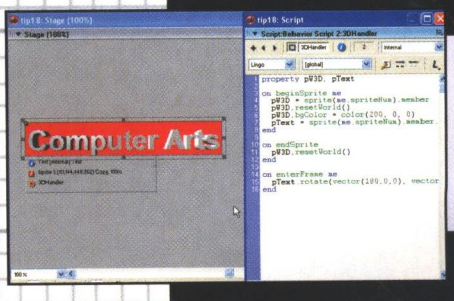
TIP 14

In *Director MX 2004* you can control *Flash* sprites with ease.


Use str = to set this string
moveToFifty called
50
Use gotoAndStop(xxx), where 1<x<100, to set the frame for the movie. The number displays the current frame number
Command1
Command2
Pressing the buttons sends a event. To respond to it create Movie script function with name of the button

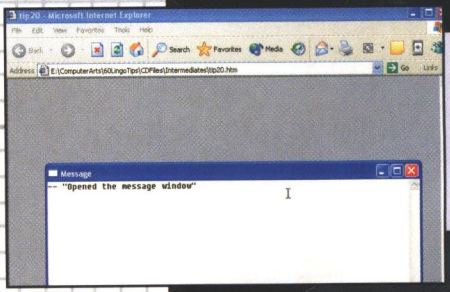
TIP 17

Call functions in *Director* from *Flash* using this code.



TIP 18

3D text is simple to create and is displayed as a 3D world.



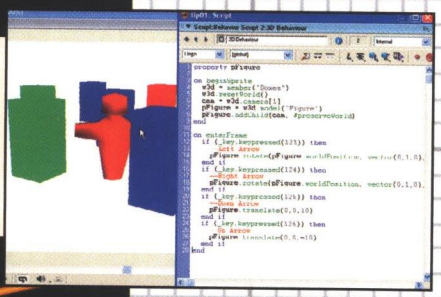
TIP 20

Test and debug a Shockwave movie within a browser window.

60 LINGO TIPS

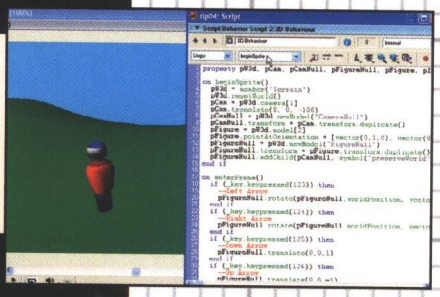
TIP 1

Using 3D Xtra, you can parent a camera to a figure.



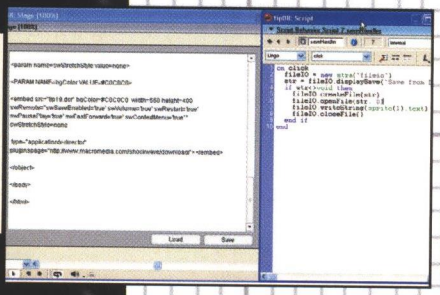
TIP 4

Rotate your object to match the angle of the terrain.



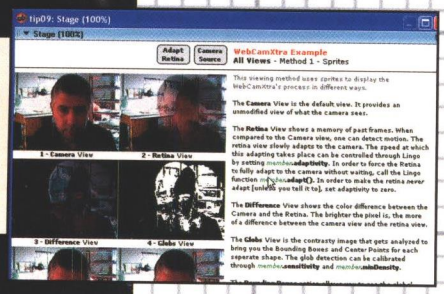
TIP 8

Using the FileIO Xtra you can create a basic word processor.



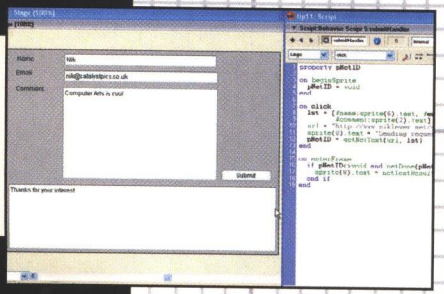
TIP 9

The WebCam Xtra is a first step to creating WebCam-sprite interaction.



TIP 11

Get data back from a PHP page using the getNetText call



1. Simple 3D camera movement

One of the most exciting aspects of *Director* is the 3D Xtra. This simple tip shows the camera following a very simple figure that moves under keyboard control. The camera is parented to the figure and so moves as the figure moves.

2. Improved 3D camera movement

Although you get movement as a result of using tip 1 (above), it doesn't have the inertia you'd expect to find in today's highly developed 3D games. The effect of inertia is achieved by using a Null object, which is an object that moves in the 3D world but has no geometry to display. You create such an object and then parent this to the figure. Then you use interpolateTo on the camera's transform object to get it to move in the direction of the camera null.

3. Basic collision detection

Collision detection is very processor intensive. Often you can use the method modelsUnderRay. This takes a position, a direction to cast a ray from (imagine a laser beam) and some options. The options can be a list, which is a new feature in *MX 2004*. This can set a maximum distance and limit the objects to be included in collision testing.

4. Following uneven terrain

If your floor isn't flat then you'll need to rotate to match the terrain orientation. This tip again uses modelsUnderRay which, when used in #detailed mode, returns the intersection position and normal — imagine a stick pointing directly out from the surface. Using the pointAt method you can rotate your figure to match the normal direction.

5. Parsing an XML string

The XML Parser Xtra is great for turning an XML document into a set of nodes or a property list. In this example, we use the XML parser to strip a text member into nodes and then display these in a *Flash* list component.

6. Accessing an XML news feed

Although parsing a cast member is good, XML really comes into its own when downloaded dynamically from the Web. News feeds are readily available and this tip accesses a news feed, then uses the XML parser to create content to populate a *Flash* list component.

7. Loading a text file with the FileIO Xtra

With the FileIO Xtra you can load a text file from anywhere on your computer or network. In this tip, we use the FileIO along with a *Flash* TextArea component to display any text document.

8. Saving a text file with the FileIO Xtra

By adding a second button for saving the file, you can create a very primitive word processor. The *Flash* TextArea component enables you to edit the content, and the FileIO Xtra looks after saving and loading the text.

9. Using the WebCam Xtra

Take a look at http://webcamxtra.sourceforge.net/ for a brilliant free Xtra. If you want to create a movie where a player can use their WebCam to interact with onscreen sprites, then this is a great place to start.

10. Creating a query string

With *Flash* components you can create an onscreen form easily. Using this form you can post data to a Web page. In this example, we send the data to a PHP page on our server. We format the data sent by first creating a property list containing variable name/value pairs. This can be formatted into a URL encoded querystring using URLEncode.

11. Getting text from a PHP page

You can get data back from the PHP page by storing the value generated by the call to getNetText. When the PHP then sends data back, the value of netDone for the

stored id will be true. Then you can access the returned text using netTextResult and the stored id.

12. Using postNetText

Another useful netLingo method is postNetText. Using this method, you can create a property list and pass both a url and the list to the method.

13. Creating 3D particle effects

You can create amazing effects using *Shockwave 3D* even without any geometry. One great technique is to create a particle effect. There are lots of settings when using particle effects and this can be off-putting. Hopefully this sample on the cover CD will get you trying your own examples.

14. Simple projectile motion

A ball flies through the air as a projectile. Its vertical position is dictated by its launch speed, the elapsed time and the strength of gravity. In this tip, which you'll find along with all the other Lingo code on the cover CD, you can see how to put projectile motion into your own movies.

15. Starting with the Havok Xtra

The Havok Rigid Body Physics Xtra is unbelievable. It's even easy to use. Unfortunately, Havok isn't part of *Director MX 2004*. If you have *MX* then just drag the Havok folder from the Xtras folder of *MX* to the Configuration/Xtra folder of *MX 2004*; if you haven't got *MX*, go to a site that uses Havok and then copy the file from your System/Macromed/Xtras folder into your program Xtras folder. Details and samples are available at http://oldsite.havok.com.

16. Using bitwise operations

As you know, numbers, pictures and movies are all stored on computers using bits. Bits are combined to make bytes (8 bits). You can combine numbers in Lingo (stored using 32 bits) using the bitwise operators. This tip shows how to use bitAnd, bitOr and bitXor.

17. Encoding using the bitwise operators

You can use the bitwise operators to encode sensitive data that you want to keep secret from prying eyes when it's being passed across the Internet. Since only you will know how it was encoded, it makes it difficult for others to decode the data. For those who know how it was encoded, decoding the number back to its original form can be done quickly and easily.

18. Converting hexidecimal to decimal

Hexidecimal uses 16 symbols rather than the familiar 10 used in decimal. The top six are the letters ABCDEF. Hexidecimal is used in HTML to define colours – #FFFFFF means white. This tip shows you how to convert a Hexidecimal string into a decimal value.

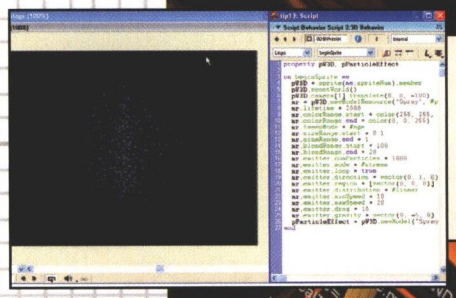
19. Bitwise operations with JavaScript

Using JavaScript, bitwise operations are more comprehensive than with Lingo. You shift bits to the left and right in addition to combining them. This tip uses the same encoding methods as tip 17 above, but the syntax used is JavaScript.

20. Calling a script function in the host Web page

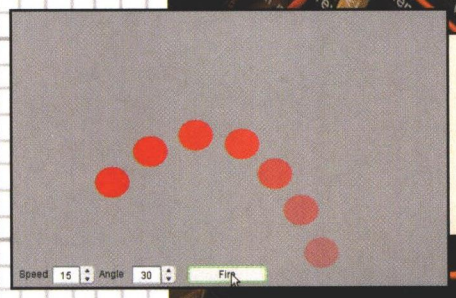
There are going to be times when you'll want to communicate with the host Web page. You can do this by using the externalEvent method. In the HTML, you need to add the following code, where tip20 is the name of the Shockwave movie in the same page:

```
<script language="JavaScript">
function
alertFunction(parameter) {
  alert(parameter)
}
</script>
<script language="vbscript">
 sub tip20_ExternalEvent(byVal
aCommand)
   call alertFunction(aCommand)
 end sub
</script>
```



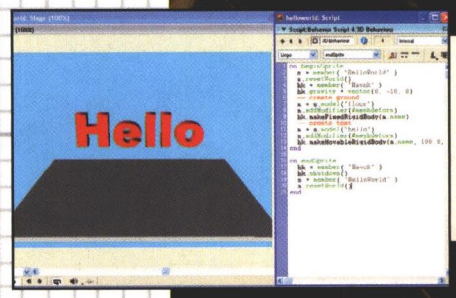
TIP 13

Create a variety of great particle effects with *Shockwave 3D*.



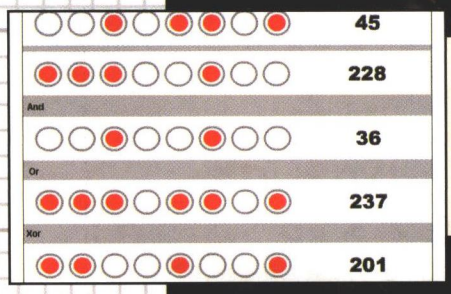
TIP 14

Find out how to incorporate projectile motion in your movies.



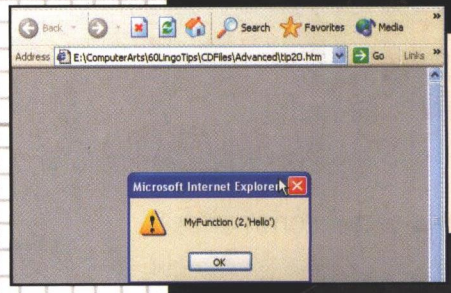
TIP 15

The Havok Rigid Body Physics Xtra can produce amazing results.



TIP 16

Use bitwise operators in *Director* for greater control.



TIP 20

Communicate with a Web page using the externalEvent method.

Mac FORMAT X

10-PAGE EXCLUSIVE

OS X 10.4 TIGER

Everything you need to know about Apple's grrrrreat new operating system!

PLUS! HOW TO...

- Master the Lasso tool in Photoshop
- Get tuneful with GarageBand
- Create huge panoramas in Stitcher
- Discover iTunes' secret commands
- Make your own logo with LogoCreator

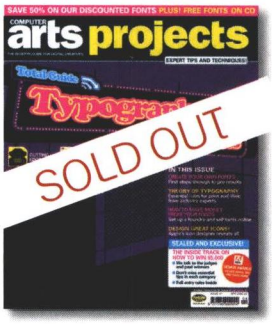
On sale Tuesday 3 August

All your Apple needs
Mac FORMAT
Packed with great advice and reviews! +2 FREE CDs
TIGER
Hands-on with OS X 10.4 – every new feature explained PLUS the stuff Apple won't show you!
+ Troubleshooting Wi-Fi
How big do you want it?
Apple's monster 30" monitor unleashed
Stitcher 3.1
Create cool panoramas from your digital photos
£39
The Logo Creator 3.5
Make your own OS X logos with this excellent utility! PLUS Free tutorials
Mac FORMAT TOOLKIT CD
PLUS
Full versions of Stitcher 3.1 & The Logo Creator worth over £400!

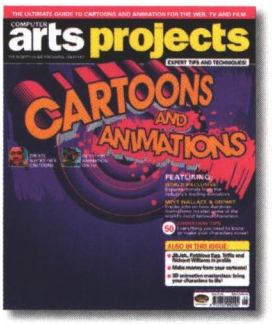
BACK ISSUES

computer arts projects

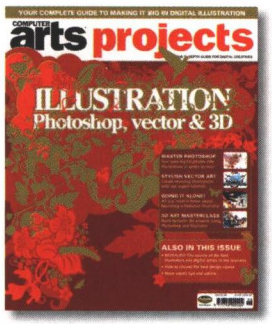
Issue 56: Perfect Figures and Faces
Code COSB0056
Our experts show you how to create fantastic figures and faces. Use *Painter* to create a digital portrait and *Photoshop* to take on the tricky business of drawing hands. And find out how to build a 3D face and figure, create lifelike hair and produce perfect skin textures.
CD Victoria 1.0 full *Poser* model, stock images worth £930 and *Illustrator CS* demo.



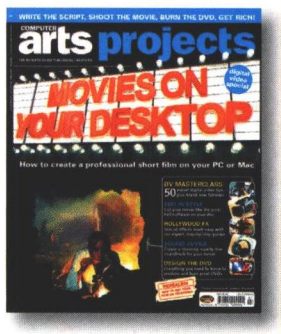
Issue 57: Typography
Code COSB0057
Expert tutorials include how to: create your own handwritten font, design display fonts, create fonts for the Web, design pixel fonts, turn type into logos, and manage your fonts with Suitcase. There are also 30 top tips on typography, help with making money from your fonts, advice on hand-crafted type and profiles of today's best typographers. Plus: 16 pages on how to win $5000 at the 2004 Adobe Design Achievement Awards.
CD Full apps: *TypeTool 1.3, ScanFont 3.2.* Plus £3400 worth of free fonts.



Issue 58: Cartoons and Animations
Code COSB0058
Your complete guide to creating cartoons and animations. For the Web: JibJab on interactive cut-out animation; Michele Garneau teaches *Toon Boom Studio*. For TV: Fictitious Egg makes *Flash* animations; Aardman-style stop animation using *iStopMotion*; Triffic Films expresses emotions. For film: 3D facial animation from Sybex; *Cinema 4D* character creation. Plus: 50 top tips and how to get a job in the industry.
CD *Plastic Animation Paper* full product, demos of *iStopMotion, Toon Boom Express* and *Flash MX* 2004.

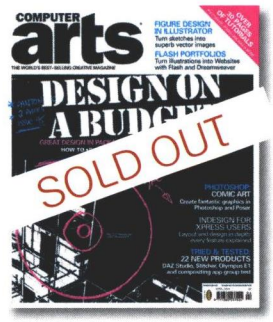


Issue 59: Illustration
Code COSB0059
Improve your illustration skills with our array of tutorials. In *Photoshop*, discover the secrets of abstract portraiture and the techniques behind Jason Cook's photo-montages. In our vector section, Identikal combines vectors and pixels, The Boy Fitz Hammond teaches character creation, and LaGraphica.com turns digital photos into vector illos. In 3D, create an abstract image and a pinball machine. Plus, tips and career advice on being a successful illustrator.
CD 50 Digital Vision illustrations, *SymmetryWorks 3.11* demo & more.



Issue 60: Movies on your desktop
Code COSB0060
Create a professional short film on your PC or Mac by following our complete project. Australian-based filmmaker Christopher Kenworthy shows you the secrets of storyboading and pre-vis, gives you 50 filming tips, explains how to edit your footage in *Avid Free DV* and lay down your sound. He also shares his knowledge of creating stunning special effects and compositing in *After Effects*, and making your titles work. Output and DVD authoring is also covered.
CD *Avid Free DV* full application, and 360MB of takes, footage and effects.

computer arts

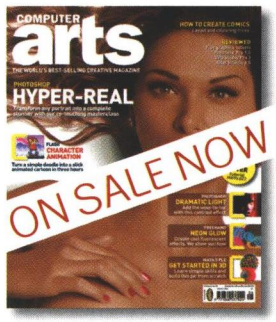


Issue 95 – Code COA95
Great designs on a budget: Website design, illustration, photo retouching and font design. Plus *InDesign* for *Quark* users.
CD *TypeTool 1.3* and *ScanFont 3.2.*

Issue 96 – Code COA96
Get your skills up to speed. Ten expert tutorials, using *Photoshop, FreeHand, GoLive, Director* & more.
CD *ACDsee 3.1* full product *and Director MX demo.*

Issue 97 – COB0097
Create graffiti cut-outs and spray paint effects in *Photoshop.* Plus, design great icons and use paths and fills to trace sketches in *Illustrator.*
CD *FontCreator 2.3* & *FONTfitter.*

Issue 98 – COB0098
Animé characters – create the cover illo. Woodblock effects in *Photoshop* and intricate vector images in *Flash.*
CD *iCorrect Entrée* full PC app, 175 *After Effects* plug-ins for Mac & PC.



Issue 99 – COB0099
Create stunning hyper-real portraits in *Photoshop.* Plus, *Flash* character animation and getting started in 3D.
CD *Realviz Stitcher 3.1* and *Maya 5 PLE*, plus a copy of *BroadbandNow!*

CALL THE HOTLINE ON: +44 (0) 870 444 8455

or TAKE OUT A SUBSCRIPTION – SEE PAGE 52

All back issues cost £6. Postage free in UK. Add £1 per issue for European orders. Add £3 per issue for the rest of the world.

TRK 002
TRK 003
TRK 004
TRK 005
TRK 006
TRK 007
Illustration by Lee Hasler [e] lee@eyeport.co.uk

Make an MP3 music player

We show you how to use *Director* Lingo to create a cross-platform MP3 player with play-list...

Macromedia *Director* is one of the frontrunners in multimedia authoring with its set of easy-to-use tools and intuitive almost pseudo programming language, Lingo. *Director* can utilise many operating system commands through Lingo or through third-party plug-ins (known as Xtras), which enable you to create cross-platform software without requiring a degree in rocket science.

Although *Director* ships with a free-of-charge FileIO plug-in, it doesn't contain all the calls required for a play-list. Buddy API is a commercial Xtra which has a huge number of functions, and the demo version enables you to use any two functions you wish for free. The demo of Buddy API can be found on this month's cover CD.

Creating a play-list for the MP3 player requires interaction between the software and the operating system to pull in files from certain folders. In Windows, the special character that's used to separate items within a file structure is the \ character. This is known as the delimiter. On the Mac, the delimiter is the : character. Instead of having to write two sets of code through a program for each system, it's a good idea to define the delimiter right at the start of the application and hold it in a variable. A variable is an object that can store an arbitrary value.

For example: global pathDelimiter

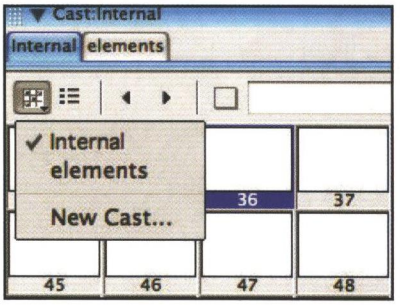
```
on doPlatform
  if thePlatform contains "MAC" then
    pathDelimiter = ":"
  else
    pathDelimiter = "\"
  end if
end
```

Expertise provided by Jon Paul Alongi. You can contact Jon at **[e]** jon@alongi.co.uk or visit **[w]** www.alongi.co.uk

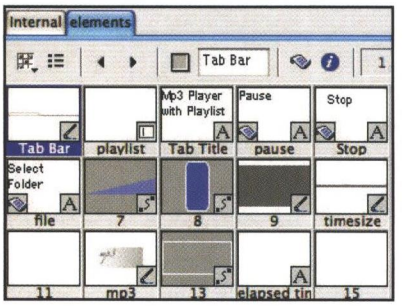
All the components needed to complete this tutorial can be found on the cover CD in the Software \ Tutorial \ Tutorial files \ All \ MP3 player.

Part 1: Setting up Director

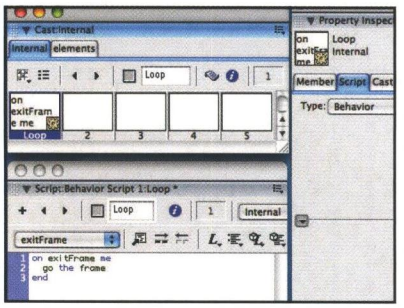
Prepare the cast before starting the action...



1 Begin by creating a new cast called elements to hold any non-script media. This helps to keep things organised.



2 In the elements cast, create text boxes to reflect the play-list, song title bar and song duration counter. Label them all accordingly. These will get referred to in the code later. Import any desired graphics and place them on the score.



3 In the Internal cast, create a new behaviour and apply it as a frame script to a chosen channel. To make the play head loop on that frame, open the behaviour and enter the code:

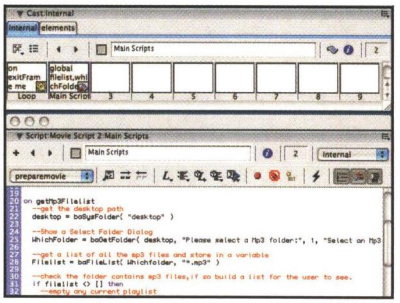
```
on exitFrame me
  go the frame
end
```

Part 2: The play-list

Compile your MP3 play-list using Buddy API...

Globals appeal

If you declare your variables as Globals, you can watch their progress in the Object Inspector. This can be extremely useful when programming.



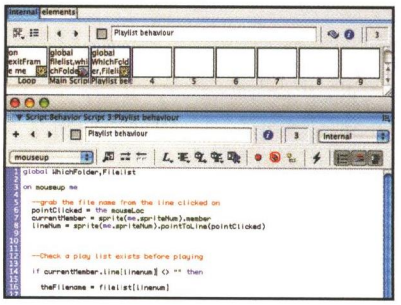
1 Create a new movie script called Main Scripts to hold the MP3 player handlers. Make sure that Buddy API is installed into your Director Xtras folder because you'll need to call its file list function.

```
desktop = baSysFolder( "desktop"
)
WhichFolder = baGetFolder(
desktop, "Please  select  a Mp3
folder:", 1, "Select an Mp3
folder", -1, 0 )
Filelist = baFileList(
Whichfolder, "*.mp3" )
```

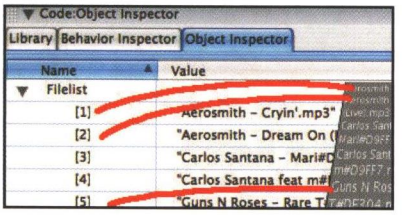
2 Create a new handler called: getMP3Filelist. You need to open a selected folder window using Buddy API, but it requires a starting path. Since you don't know which platform your end user will be using, you can ask Buddy API to tell you the path to the desktop. Buddy API can then return a list of all the desired files in the chosen folder.

```
if filelist <> [] then
  member ("playlist").text = ""
    repeat with i = 1 to
filelist.count
member("playlist").text =
member("playlist").text &
filelist[i] & RETURN
end repeat
else
nothing
end if
```

3 The returned files are in *Director's* List format. If there are no files, the list will look like this: []. It's good practice to check that files do exist otherwise you're likely to get errors later on. You can then move on to appending the list into your play-list text box using the code above.



4 To interact with the file list, you need to add a behaviour to its sprite. Create a new behaviour script and label it Playlist behaviour. Drag it on top of the play-list sprite to initiate it, then open the script.



5 Each line number represents the position of each MP3 in our list. In the MouseUp handler you can grab the line number by entering:

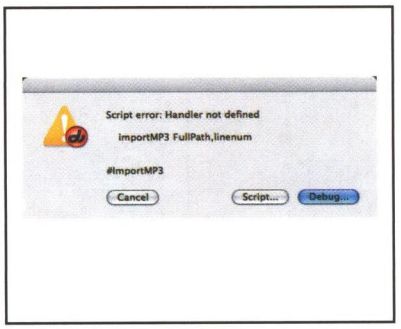
```
global filelist,whichfolder
On MouseUp me
pointClicked = the mouseLoc
currentMember = sprite(me.spriteNum).member
lineNum = sprite(me.spriteNum).pointToLine(
pointClicked)
```

```
if currentMember.line[linenum]
<> "" then
theFilename = filelist[linenum]
FullPath =
WhichFolder&thefilename
importMP3  FullPath,linenum
else
    nothing
  end if
end
```

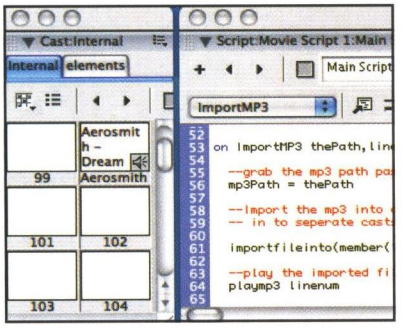
6 You need to check whether the user clicked on a line in use before trying to use the information. Once you have your result, you can build the data into a file path and pass it to the executing handler.

Helpful messages

The Message window can be extremely helpful while scripting. If you haven't already made calls to your handlers, you can execute them in the Message window and also view any results returned.

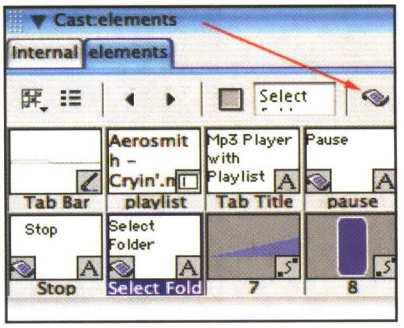


7 You'll probably get an error at this stage, because you're trying to call the handler importMP3 which doesn't exist yet. We chose to put this in a separate handler, because later on in the program we may want to do something similar and we don't want to have to duplicate code.



8 Open your Main Script and import the MP3 with the following handler and code:

```
on ImportMP3 thePath,linenum
mp3Path = thePath
importfileinto(member(100), mp3Path)
end
```



9 Create a button for opening the Folder dialog box. Click on the Cast Member Script button and enter the getMp3Playlist command in the MouseUp handler.

Part 3: Controlling the MP3

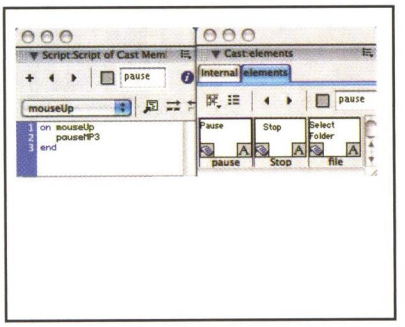
Enter the code needed to get your MP3 player working...

```
on ImportMP3 thePath,linenum
mp3Path = thePath
importfileinto(member(100),
mp3Path)
playmp3 linenum
end
```

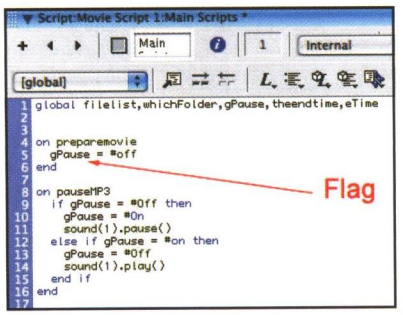
1 To play the MP3 chosen from the play-list, first insert this code into the ImportMP3 handler: `playmp3 linenum`

```
on playmp3 linenum
  sound(1).play([#member:
member(100,"internal"),
#startTime: 0, \
 #loopCount: 1, #loopStartTime:
0, #loopEndTime: 0])
end
```

2 The program will throw up an error because the handler does not exist yet. Type the above code into the Main Script. Running the program will now enable you to play MP3s via the play-list.



3 If you haven't already created Pause and Stop buttons, do so now in the elements cast. In turn, open their Cast Member Scripts and enter the command to initiate the function. We've chosen PauseMP3 and StopMP3.



4 Pausing the MP3 is as simple as entering: Sound(1).pause. To unpause, you can set a property known in programming as a Flag. This is a variable which will represent the state of the action you want to perform. In the Main Script create the handler: on preparemovie. Enter: `gPause = #OFF`

```
global gPause
```

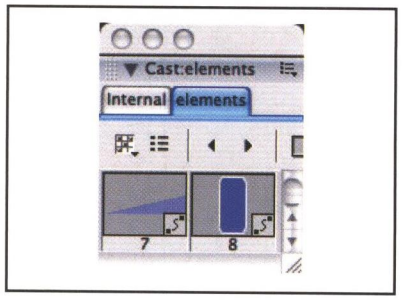
5 Since we want to interact with the gPause value in other handlers, we must set it as a global. This also means we can track its value in the object inspector. Enter the global code above.

```
on pauseMP3
  if gPause = #Off then
    gPause = #On
    sound(1).pause()
  else if gPause = #on then
    gPause = #Off
    sound(1).play()
  end if
end
```

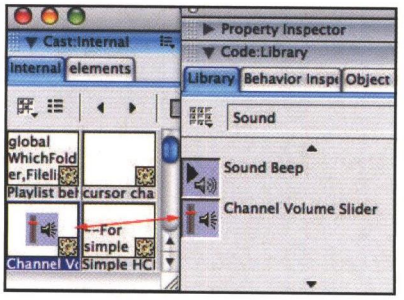
6 Since the flag is set to Off at the beginning, we can say: if the flag gets turned on, pause the MP3; but while it's off, play the MP3. Enter the code above into the Main Script.

```
on stopmp3
  sound(1).stop()
end
```

7 Enter the code above to stop the MP3. This will return the MP3 to the beginning, unlike Pause, which keeps track of the time index.



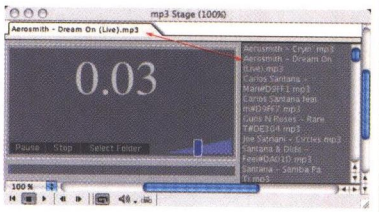
8 Now you need to create two graphics to represent a volume bar and slider.



9 Navigate through the Library palette to Media: Sound and drag the Channel Volume Slider into your Internal cast. Place the volume graphics on the stage and drag the volume behaviour on to them.

Part 4: Validation

Validation shows the user what they've done, what they're about to do or what is happening...



1 Inside the playMp3 handler, add the following code to change the tab title bar to the MP3 file name:

```
member("tab title").text= filelist[linenum]
```

```
on stopmp3
sound(1).stop()
member("tab title").text= "Mp3
Player  with  Playlist"
end
```

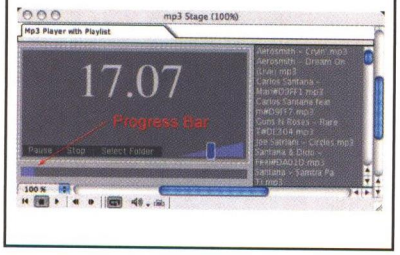
2 To make the title revert to its original name, you must set it in the stop handler.

```
the floatprecision = 2
member("elapsed time").text =
string(sound(1).
elapsedTime/1000)
```

3 *Director* outputs the elapsed time of the MP3 in milliseconds. You need to divide the number by 1000 and adjust the floating point for a more readable display. Create a new handler in Main Script called TimeDisplay and enter the code above.

```
if soundBusy(1) then
    timedisplay
  else
    nothing
end if
```

4 You have to call this handler, but only when the sound is playing. You can do this by refining the Loop behaviour created earlier and check to see if the sound channel is in use. Append the code to the loop handler.



5 A progress bar is a good visual user validation, which can be done using Imaging Lingo as part of the TimeDisplay handler. The first step is to create a bitmap that's one pixel in size. Label it TimeSize.

```
iwidth = 340
iheight =14
```

6 Since TimeDisplay is called every time the frame is exited, you can create a new progress bar according to the percentage of elapsed time from the total. At the beginning of the handler, set two variables to hold the desired height and width of the bar.

```
Percent =  sound(1).elapsedTime/
sound(1).endTime
NewWidth = float(iWidth/percent)
```

7 Before you start creating the bar, it's good to get all the values you need first. Work out the percentage change in time and save in a variable. Apply the percentage to the desired width and put in a new variable.

```
newimage =
image(iwidth,iheight,32)
newimage.fill(rect(0, 0, iwidth,
iheight),rgb(143, 143, 143))
```

8 We can now create the whole progress bar image in memory according to initial data.

```
newimage.fill(rect(0, 0,
newwidth, iheight),rgb(153, 153,
255))
member("timesize","elements").
image = newimage
```

9 Using the new width calculated from a percentage, you can append a different colour on top of the bar to show the progress.

Part 5: Tidying up

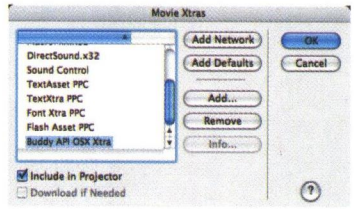
Take these steps to completing your MP3 player...

```
 member("playlist").text= ""
  member("tab title").text= "Mp3
Player  with  Playlist"
  member("elapsed time").text
=""
```

1 Some text boxes are holding different messages according to the code. You need to make sure the program starts off with the correct data by appending the preparemovie handler with the above.

```
on mousewithin  me
  sprite (me.spritenum).cursor =
280
end
on mouseleave  me
  sprite (me.spritenum).cursor =
-1
end
```

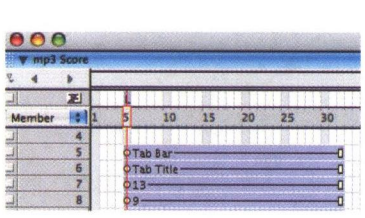
2 Changing the cursor is often a good way to show the user validation. Simply create a new behaviour script and enter the code above. Drag the behaviour onto any sprite you wish to affect.



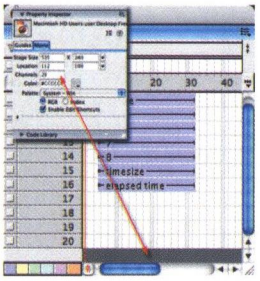
3 Since you're using the Buddy API Xtra, you'll need to add the Xtra to the Xtras list so that *Director* can incorporate it into the final projector.

Part 6: Final considerations

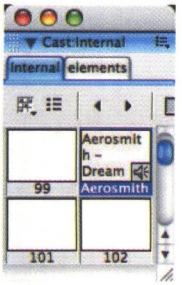
There are three more small things to bear in mind...



1 *Director* tends to start better when sprites are not placed in the first channel. This is due to the beginsprite handler having issues in that channel. It's good practice to avoid doing this.

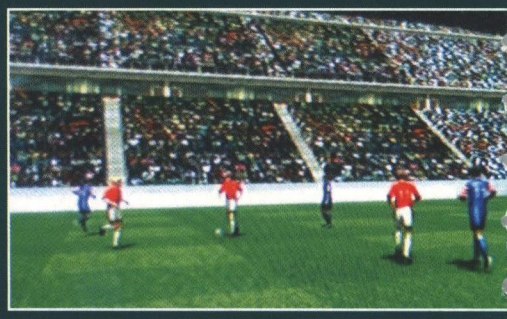


2 File size is always an important issue in authoring. Trimming the amount of sprite channels can help to reduce wasted space.



3 If you've been testing the application within *Director*, member 100 will have acquired an MP3. You should ensure you delete this member before you compile your final application. ca-p

THE MAKING OF Virtual Replay



Shockwave 3D has enabled the creation of a new and dynamic product that puts online viewers in the place of any player on the pitch and replays the action from their eyes...

At the time of writing, England has just tumbled out ignominiously from the Euro 2004 tournament, followed swiftly by France, Sweden and Denmark, so by the time this feature comes out, it may dredge up some rather painful memories for a sizeable percentage of the UK. Presently though, the matches are on and the BBC Sports Website is buzzing with heightened activity. This is due mainly to the rather nifty Virtual Replay service, a way to revisit or catch up on the silver goals and goofs almost as soon as they happen.

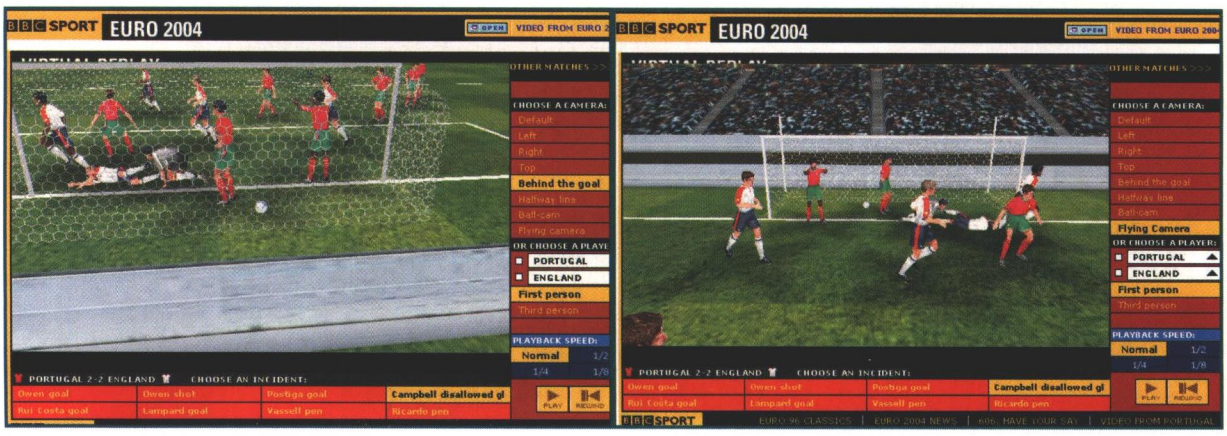
The BBC Sport Website (news.bbc.co.uk/sport) features a conventional media player to view video highlights of the games from Portugal, but the Virtual Replay service provides a way to replay key events from the tournament, presented in animated 3D 'toon form. You can see every goal or 'match incident' from a number of different camera angles as well as from the perspective of the key players. You'll need the Shockwave plug-in version 10 or later to access it, but once you navigate to the page at www.bbc.co.uk/virtualreplay/ you are presented with a chart of all the Euro 2004 tournament matches.

Covering all the angles

Simply choose the match you want to revisit and the first animated incident loads automatically after a quick whisk of a progress bar on broadband or a three-minute download on dial-up. You can then choose any number of incidents and play them back from first- or third-person perspectives of various players, from behind the goal or from a ball view. Users can also change playback speed (slowing down to an eighth of normal) and have the option of rewinding the action to play over. It all makes for addictive viewing.

Virtual Replay is the fruit of a union between 3d web technologies (www.3dwebtech.co.uk) and the Russian Web Service, with the current Euro 2004 project designed and incorporated for the BBC by BBC Sport Interactive. "The main purpose of Virtual Replay is to reconstruct an incident and give new analysis tools to the viewer," says Michael Kossoy, Creative Director of Russian Web Service, which is based in St Petersburg. "We've worked with interactive 3D graphics for many years and our intention was to apply our 3D knowledge and experience to a new area, namely sports."

Working with Cheshire-based 3d web technologies, the team created a prototype proof of concept and approached BBC Sports. "It's the kind of thing we'd been looking at doing for a couple of years now, something along the lines of an online replay, but this is the first time we've come across something that we wanted to connect to," says Ben Gallop, Editor of BBC Sport Interactive. What we were looking for is an added dimension. We're the biggest sports site around in terms



Let's see that again

Turning around unforeseen movements quickly...

So just how does Virtual Replay work and how on earth does the team manage to turn the animation round so quickly? "There are three constituents: a big library, good interfaces and training," says Michael Kossoy. "We developed a big library of *MagicTrack* movements specific to football, which we thought would be enough for any eventuality, but Euro 2004 proved us wrong! We didn't imagine goalkeepers hanging on a bar, players grabbing opponents and throwing them on the grass or even spitting in their faces. So our library became far more lifelike during Euro 2004.

"Our specialised software modules also feature very good interfaces," continues Michael, "so we don't lose time when accessing player animation properties – everything is at hand. Another key point is training. Initially we spent four days producing one goal, but now we have reached the minimum of four hours.

"Working night and day on the job makes this project really one of a kind, but currently there are 'white nights' in St Petersburg, where we have the bulk of the production team. It doesn't get dark during the night, so it is relatively easy to go without sleep!"

of market share and we pride ourselves on rigorous journalism, but the one area where we wanted to develop was that of innovative features. We wanted to show something that was more than just a text analysis of sports, something that really plays on the strength of the Internet. I think this does exactly that."

Rooney magic

You may think that there's some motion-capture or similar arcane jiggery-pokery involved in replicating Rooney and co on the field of dreams (or sorrows) but in fact the Russian Web Service and 3d web technologies used custom-built software called *MagicTrack* to develop Virtual Replay. "*MagicTrack* draws from a library of movement to output a set of motion tracks for the individual players," says Andy Dennison, Director of 3d web technologies. "Initially this data was loaded into a custom-developed plug-in for discreet *3ds max* and used to apply animations to players. The result is saved for the Web and can then be used with our Shockwave player."

"The technology development started in the second half of 2002," says Andrey Piatlin, Director General of Russian Web Service. "The first version didn't require any special hardware, but the visual quality was not so good. We initially used Blaxxun3D (a Java applet) but then selected Shockwave for 3D visualisation for two reasons: it supports 3D acceleration and it is well-known and widely used. The current version offers far better visuals, especially when you have a 3D graphics card. We managed to bring down the time to develop each virtual replay to 4-6 hours."

The initial concept, though innovative, had to pass some rigorous testing before being blessed with the BBC brand. "They had some examples that they showed us, which were quite ambitious, and we were incredibly impressed with the turnaround time of a few hours, but you always have to possess a dose of healthy scepticism," explains Ben Gallop. "We really wanted to see it produced properly before the tournament began, so we tried them out with a couple of goals from Euro 96. We gave them a challenge to produce a Gazza goal and an Alan Shearer goal and they completed it – by that time we were pretty convinced it would be a success."

The technical bits

Bolstered by the success of the trial, development for the latest version of the software was finished just before Euro 2004. "We could already reconstruct the events and offer some analysis tools, such as intriguing points of view, offside line and distance markup," recalls Michael Kossoy. "We plan to introduce other new analysis tools with each new version. Our initial prototype used *3ds max*, but we've now extended the capabilities of our *MagicTrack* software to use it for the output."

Each match animation is around 300kb in size and is presented in a specially designed Shockwave front end. "When they came to us, it was very much an off-the-shelf product," says Ben Gallop. "After we first saw VR we asked for it to be changed from a Java-based application to Shockwave. It was up to our design team to produce a page that would fit in with our style, and our Design Manager James Howard tweaked the design of the application so that it reflected the BBC Sport branding. There's a lot of work that's been done behind the scenes at our end, involving editing the clips to be used for Virtual Replay. We also now use a version of the service in live studio broadcasts during *Match of the Day*. The output is converted to a QuickTime movie, specifically for use in broadcast. This in turn is ingested into an edit suite and then output for TV to provide on-screen analysis of the big games."

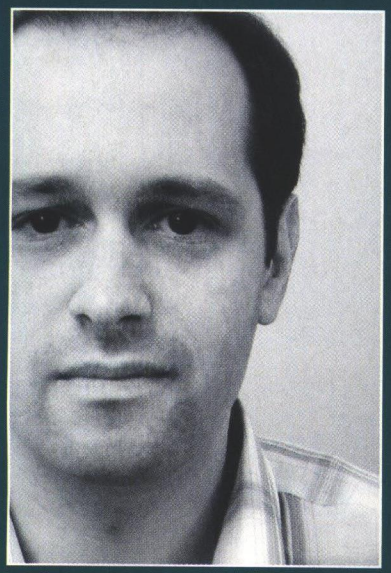
Virtual Replays: The future

Reliving the moment on a big screen...

"We want to give viewers a very intuitive and useful interface that provides them with information," says Andrey Piatlin. "The most enjoyable experience is to produce virtual replays just after real incidents occur. People are able to see replays the next morning, so we are really chasing events, not as fast as television but still quickly enough. This makes us feel part of the events, we feel as if we really were there. That is a new interface, a new way of sharing presence."

To further this aim and brand the Virtual Replay concept, his St Petersburg based outfit has now joined forces with 3d web technologies in a new joint company, LifeInMedia (www.lifeinmedia.com/). In future versions, it's clear that the team would prefer to run the replays in a bigger window. Constrained by the design of the BBC Sport website, the 3D window size is at present limited in terms of width and height to 600x340. "Viewers now have really wide screens so we would like at least to have a separate version for those who want to see it big," says Michael Kossoy.

Half-time commentary



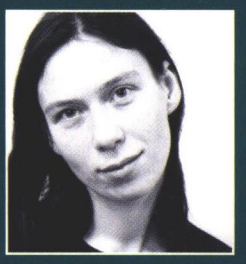
We managed to drag some of the team involved in Virtual Replays away from the screens to give us some quick quotes on their project...

Andrey Piatlin, COO, lifeinmedia

"Virtual Replays of Euro 2004 turned out to be the most exciting project ever for our team. We really like to see the result of our work posted almost immediately on the BBC Sport Website, because so many people access it. That is all an artist needs in this world! The ideas of Michael Kossoy, our Creative Director and the author of Virtual Replays, forced all of us to love football. I have seen only one televised match of Euro 2004, when I took a night off – I saw the rest only on Virtual Replays."

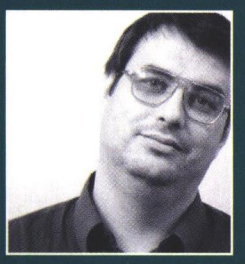
Andy Dennison, Director of 3d web technologies

"We plan to improve analysis tools and go for other sport events. We have had a lot of interest from all types of media groups around the world interested in using this technology not only for football but just about every other sport you can think of, from baseball to the pole vault."

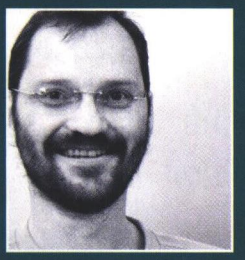


Nadezhda Zakharevich, Core Developer, lifeinmedia

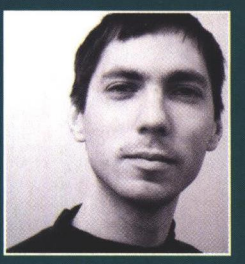
I'm not a football fan... but I love virtual reality. I'm happy every time we succeed in developing not just a raw virtual reality based product, but also a product which really makes sense and fills a need. It is great to give football fans more opportunities to touch the action."



Anton Vasiliev



Igor Perevoznikov



Stas Grozdilov, Lead Designer, lifeinmedia

"Virtual Replays give all sport fans the unique chance to see the game from the eyes of the footballers, to envisage the high points of the sport victories or suffer the downfalls of defeats. What can be more exciting to a football fan than to be right in the middle of the action?"

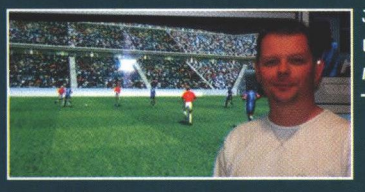
BBC Sports Interactive

Ben Gallop, Editor, BBC Sport Interactive

"Anecdotal feedback has been fantastic – people have been raving about it to their friends – and the traffic has been impressive too. The Virtual Replay service has been a huge success and we're really pleased with the way that it's gone. It's an opportunity for us to do something really innovative. With a big event like this already bringing people to the site in droves, it's a good time to showcase the system and develop a bit of viral interest as well. The feedback we've received is that word is getting out there. We're keen to use this system for other sporting events and indeed other sports. There are issues we have to look at, such as rights, but I'm confident we can keep Virtual Replay going for other big events."

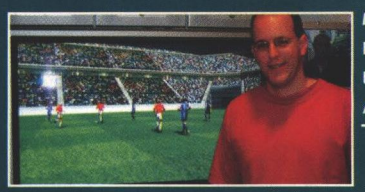
Helen Dapre, Broadcast Journalist

Karen Rollins, Broadcast Assistant



James Howard, Design Manager

Neil Hall, Senior Broadcast Journalist



Martin Richardson, Broadcast Assistant

NEXT MONTH
COMPUTER arts®
100th ISSUE
ON SALE THURS 5TH AUGUST

COMPUTER arts

MISSED AN ARTICLE?

BUY TUTORIALS FOR JUST 99p!

No more back issues left? Don't worry! You can download previous *Computer Arts* tutorials online with our new, secure, one-stop shop. Just go to www.computerarts.co.uk/tutorials

SO WHAT DO I NEED?

All tutorials are in PDF format. You'll need Adobe *Acrobat* to open them, which you can download ***free*** from www.adobe.co.uk

Once you've paid for your tutorial, you'll have 24 hours to download the file. You can then download all supporting files for the tutorial via a link embedded in the PDF

click & buy from BT

BT CLICK&BUY

BT click&buy is a secure way to pay for online content. If you have a BT telephone, then you can have the 99p charge added to your phone bill. If you don't have a BT account, simply charge it to your credit card instead. Both options are safe and secure ways to pay.

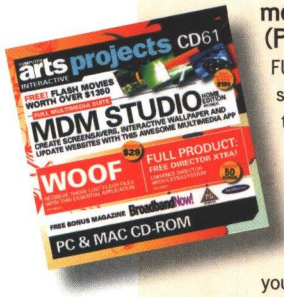
Go to www.computerarts.co.uk/tutorials

COMPUTER arts projects INTERACTIVE

CD Contents

This month's line-up of software, *Flash* movies and tutorial files

CD CONTENTS AT A GLANCE



mdm Studio Home Edition (PC only)

FULL PRODUCT: Use this fantastic suite of multimedia authoring tools to create fantastic screensavers, training videos and more! Originally sold for $199.

Woof (PC only)

FULL PRODUCT: Worth $29, you can use this extremely handy tool to retrieve lost *Flash* files on your system.

cXtraSysTray (PC only)

FULL PRODUCT: Use this full *Director* Xtra to create an application icon in a Windows user's system tray. Exclusive Xtra worth 50 EUROS.

Bigshot Media Flash movies

FREE FLASH MOVIES: Worth over $1300, you can use these 15 *Flash* stock movies in your own commercial projects.

Killersound sound effects

FREE SOUND EFFECTS: Add life to your multimedia projects with these samples from the extensive Killersound audio library.

mediamacros behaviours

DIRECTOR BEHAVIOURS: Check out this assortment of 84 high-quality *Director* behaviours from our good friends at mediamacros.

FlashJester demos

13 DEMOS: Take all of FlashJester's range of *Flash* add-ons for a test-drive. Demos include Jmovie, Entertainor and Creator. See your CD for full details.

Director MX video training

EXCLUSIVE: Brand new video training from Lynda.com. Get to grips with the new version of *Director* with this excellent training series.

BroadbandNOW!

FREE: The world's first paperless magazine – *BroadbandNow!* See **[w]** www.broadbandnowmag.com. Parental Guidance. General viewing but some scenes may be unsuitable for young children.

PG

CFJ09169

mdm Studio Home Edition

Create fantastic content with this multimedia authoring app!

mdm Studio Home is a suite of four Web tools that enables you to take the functionality of Macromedia *Flash* a little bit further. Using this app you'll be able to create fantastic applications, training videos and *Flash*-based wallpaper for PC users.

This application has previously sold for £150 and is for use on personal projects only, but you can nevertheless create some stunning projects with the suite. See below for details of the individual tools.

Flash Studio Home

This tool enables you to compile a fully customisable projector file (.exe) or Windows Screensaver (.scr) from your *Flash 4, 5* or *MX* SWF files. You can create organic-shaped .exe files – giving your applications varied and interesting window shapes. You can also extend *Flash* ActionScript using fscommands, and pack five additional files into the .exe application you create.

FlashCast Home

FlashCast is the ideal tool for creating training videos and learning tools in the SWF format. It captures your desktop activity, including mouse clicks and key presses and then you simply output the result as an SWF. You can alter the output by using the spreadsheet-like interface, giving you excellent control over your training content. Essential for teaching colleagues and clients new tricks!

FlashCandy Home

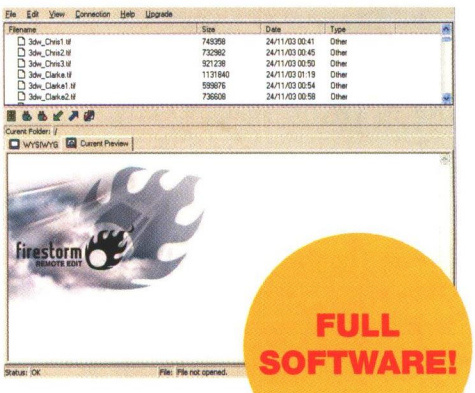
FlashCandy Home is a tool for creating interactive, *Flash*-based wallpaper for Windows machines. You can add multimedia content to your user's desktop in no time at all! The app gives you the ability to create an installer for the wallpaper, as well as giving you a WYSIWYG preview of the wallpaper before you create it.

FireStorm Remote Edit Home

FireStorm Remote Edit Home enables you to connect to any Website on the Internet via FTP, and edit or update it in a WYSIWYG environment. You can make changes as if the site is on your HD. With this Home edition of the software, you can upload images, as well as make spelling changes and so on. It's a great tool to have in your creative armoury.

· To get 25-per-cent off the latest version of *mdm Studio*, go to **[w] www.multidmedia.com/shop** and enter the following promotion code: **Caupg**

· See **[w] www.multidmedia.com** for more. Please note you need an Internet connection to install this product.



FULL SOFTWARE!

Also on the CD...

You'll find all the files and code you need to follow this month's tutorials in the Software/Tutorial files folder on your cover CD. We've also included all the Lingo and ActionScript code for our 120 tips.

FULL SOFTWARE!

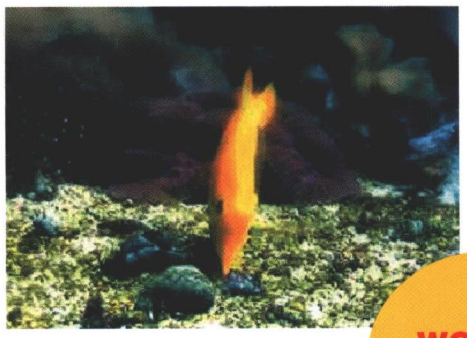
Woof

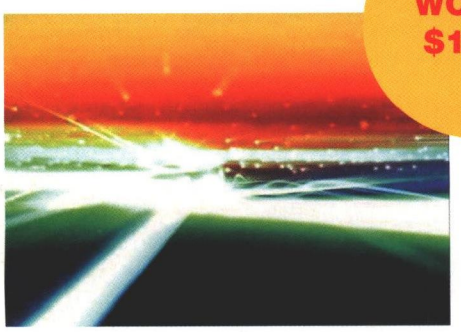
Retrieve those lost *Flash* files with this nifty application worth $29...

Meet *Woof*, your faithful companion for *Flash* retrieval. This unbelievably handy application, as sold for $29, will search your local drives, Netscape or Internet Explorer looking for *Flash* files, Real Audio files and JPEGs. It then brings you a list of what it found, enabling you to browse through the sites in your own time.

It's an ideal app for searching for that lost piece of inspiration, and thanks to *Computer Arts Projects* and FlashJester, it's yours for free. There are also 13 demos of FlashJester's range of *Flash* add-ons on the cover CD.

- To install *Woof* use the following serial number information: Username: **Computer Arts Reader**, Serial number: **5590615B**
- Find out more about FlashJester's range of tools at **[w] www.flashjester.com**.





WORTH $1369

Bigshot Flash movies

15 free movies for you to use in your multimedia projects...

Bigshot Media Inc. is the world's first resource of stock *Flash* movies. These 15 free movies are FLA files, which you can customise to bring live action imagery to your site, Web ads or new media projects.

These are for use in any commercial or personal project and would usually cost you over $1350! There's everything from underwater to abstract motion graphics included here – so try them out for yourself and be sure to check out the rest of Bigshot's collection...

- Find out more about Bigshot's range of *Flash* stock movies at **[w] www.bigshotmedia.com**.

Flash and Director trials

Download the latest 30-day trials of *Flash* and *Director* now!

If you don't have *Flash* or *Director*, don't worry, you can download 30-day trial versions from Macromedia's Website. Simply follow the link below to get your hands on the demo versions, and then you can get cracking with all of our tutorials... **[w] www.macromedia.com/software**.

NEXT ISSUE IN COMPUTER arts projects

BECOME A WEB DESIGN SUPERSTAR

Create a fantastic Website in Dreamweaver or GoLive using our expert guide...

· PHP explained!
Hook up your site to a database and add interactivity! We explain exactly how PHP works and how it can benefit you.

· The secrets of JavaScript
Learn how this programming language works and how you can use it to add extra elements to your site designs.

· CSS for everyone!
The secrets of Cascading Style Sheets revealed! Why you should be using these to format your Website pages easily.

· Beginning with Dreamweaver
The basics of *Dreamweaver* explained. We guide you through the first steps to creating an amazing Website.

· Get started with GoLive
Learn how to combine *GoLive* with the rest of Adobe's CS tools to make a compelling, media-rich site.

· Using Smart objects
Link *Photoshop* and *Illustrator* files to your *GoLive* pages and never worry about optimising and updating images again. Step-by-step tutorial.

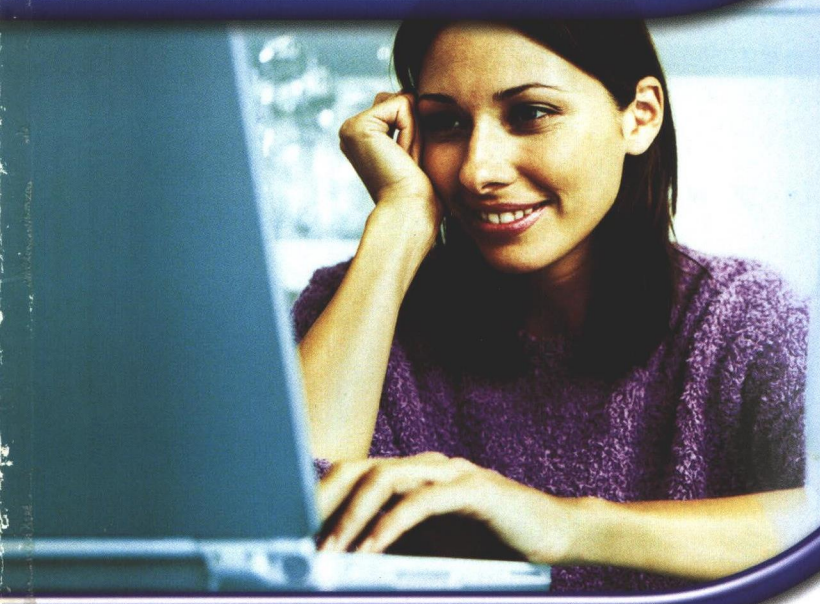
· Top HTML tips
Essential techniques for optimising your code and keeping file sizes low — never leave your site visitors waiting for downloads again!

· Design for Broadband
With Broadband now available to almost everyone, you should be thinking about how to improve the multimedia content of your site. We show you how.

All contents correct at time of going to press

Computer Arts Projects 62 • ON SALE Thursday 19 August 2004!

123-reg.co.uk



EXCLUSIVE OFFER

Transfer-in your domain name for FREE!*

There's no better place to host your website

Starter

only £1.59† per month

- 100MB of web space
- 1500MB data transfer (bandwidth) per month
- CGI/Perl Script enabled
- On-line file manager and full FTP access
- 15 mail boxes (POP3) • Free WebMail
- Webalizer graphical web statistics
- Add FrontPage Extensions for £25 per year
- Unlimited e-mail forwarding addresses

Starter Plus

only £2.99† per month

- 250MB of web space
- 3000MB data transfer (bandwidth) per month
- CGI/Perl Script enabled
- On-line file manager and full FTP access
- 25 mail boxes (POP3) • Free WebMail
- Webalizer graphical web statistics
- Add FrontPage Extensions for £25 per year
- Unlimited e-mail forwarding addresses

Starter Pro

~~£4.99~~ £3.99†▲ per month

- 500MB of web space
- 5000MB data transfer (bandwidth) per month
- CGI/Perl Script enabled
- On-line file manager and full FTP access
- 35 mail boxes (POP3) • Free WebMail
- Webalizer graphical web statistics
- Add FrontPage Extensions for £25 per year
- Unlimited e-mail forwarding addresses

It's never been easier or cheaper to have your own domain name

Register

only 9p†† UK Domain Names

123-reg delivers a totally automated service with the lowest prices guaranteed!

This exclusive package includes:

- Web based control panel • Online domain name management system
- Unlimited email forwarding aliases • Catch-all e-mail forwarding • Unlimited web forwarding
- Manage name servers • Affiliate scheme • Home page site wizard
- Manage your own DNS • No transfer out fee • Plus much, much more!

*Offer valid for .co.uk domains and .com domains are less than half price with an extra years free registration added. Offer ends 31st August 2004.
▲Promotion extended to 31st August 2004.
†Monthly fee must be paid annually and excludes a one-off setup fee of £9.99.
††Plus registrar registration fee, see website for full details. All prices exclude VAT. E&OE.

Host Europe

www.123-reg.co.uk

juice drops

LAYERED
PHOTOSHOP BACKGROUNDS
IN THREE COMPLETE COLLECTIONS

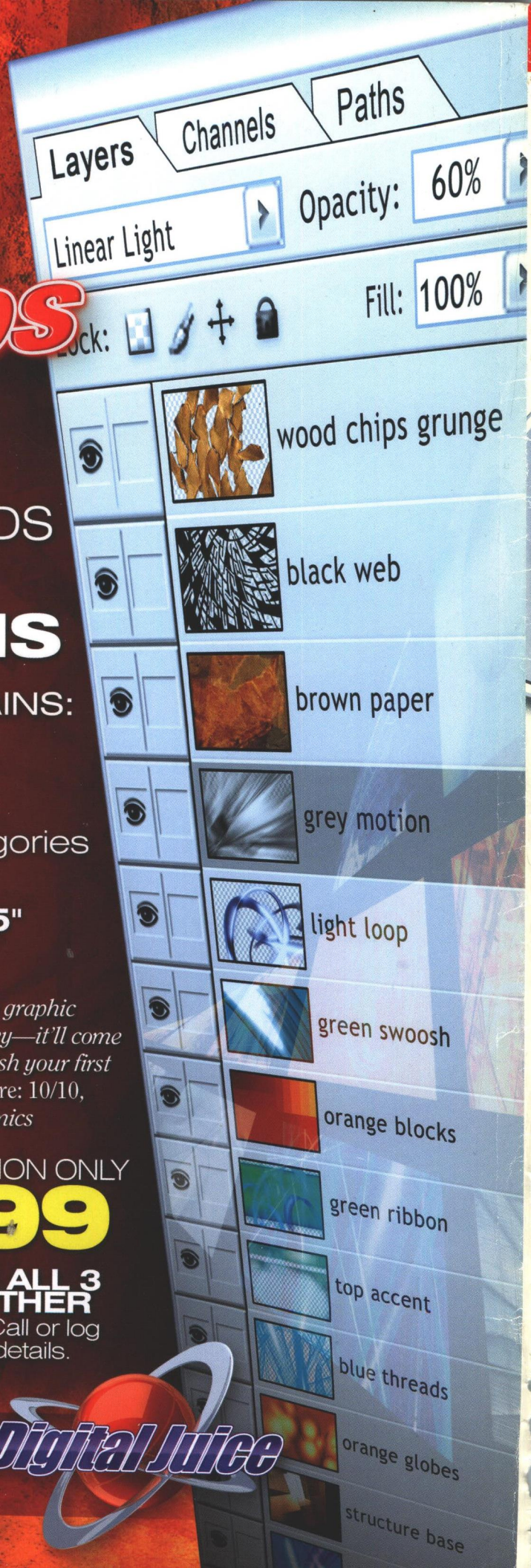
EACH COLLECTION CONTAINS:

- **24** DVD-ROMs
- **9,000+** Layers
- **8** Distinct Categories
- **800** Images (**17.25"** x **11.25"** at 300 DPI)

"Flat out—if you're a graphic designer... invest the money—it'll come back to you after you finish your first project."—Review Score: 10/10, *NextGen Electronics*

EACH COLLECTION ONLY
US$399

OR
BUNDLE ALL 3 TOGETHER
AND SAVE! Call or log on for more details.



www.digitaljuice.com
OR CALL 800.525.2203 (U.S.)
email: sales@digitaljuice.com

Digital Juice